Calhoun
Institutional Archive of the Naval Postgraduate School

Calhoun: The NPS Institutional Archive

DSpace Repository

Theses and Dissertations | 1. Thesis and Dissertation Collection, all items

1992

Motion planning for rigid body robots

Tan, Liek Foo

Monterey, California. Naval Postgraduate School

http://hdl.handle.net/10945/23971

Copyright is reserved by the copyright owner

Downloaded from NPS Archive: Calhoun

http://www.nps.edu/library

Calhoun is the Naval Postgraduate School's public access digital repository for research materials and institutional publications created by the NPS community. Calhoun is named for Professor of Mathematics Guy K. Calhoun, NPS's first appointed -- and published -- scholarly author.

Dudley Knox Library / Naval Postgraduate School
411 Dyer Road / 1 University Circle
Monterey, California USA 93943

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b. RESTRICTIVE MARKINGS
2a. SECURITY CLASSIFICATION AUTHORITY	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited.
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Computer Science Department Naval Postgraduate School 6b. OFFICE SYMBOL *(if applicable)* CS	7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c. ADDRESS *(City, State, and ZIP Code)* Monterey, CA 93943-5000	7b. ADDRESS *(City, State, and ZIP Code)* Monterey, CA 93943-5000
8a. NAME OF FUNDING/SPONSORING ORGANIZATION 8b. OFFICE SYMBOL *(if applicable)*	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c. ADDRESS *(City, State, and ZIP Code)*	10. SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO. / PROJECT NO. / TASK NO. / WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*

MOTION PLANNING FOR RIGID BODY ROBOTS

12. PERSONAL AUTHOR(S)

Tan, Liek Foo

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT *(Year, Month, Day)*	15. PAGE COUNT
Master's Thesis	FROM 09/91 TO 06/92.	June 1992	131

16. SUPPLEMENTARY NOTATION

The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

17. COSATI CODES			18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
FIELD	GROUP	SUB-GROUP	Motion Planning, Path Planning, Rigid Body Robot, Non-holonomic, Kinematic Constraints, Cubic Spiral, Layered Motion Planning Approach, Spine Net, Sensor-Oriented Method, Yamabico

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

Given a non-holonomic disc robot $\mathcal{D}$, its motion constraints in terms of maximum curvature (κ_{max}) and rate of change of curvature ($\dot{\kappa}_{max}$), a set W of rectilinear polygonal obstacles which assemble an office-like environment, and two configurations S and G in $free(W)$, this thesis investigates the planning of a smooth free path which satisfies the following condition: $\mathcal{D}$ is allowed backing up motions at the end portions of the path, but the middle portion is to be of class C^2 in its entirety. Although the motion planning problem of $\mathcal{D}$ amidst polygonal obstacles has been extensively studied, the paths considered are mostly class C^1 and piecewise C^2 only, and are subject only to the κ_{max} constraint. Typically, such paths consist of straight line segments and circular ars which have curvature discontinuity at the junction points. In order for $\mathcal{D}$ to follow such paths physically, $\mathcal{D}$ has to stop abruptly at each junction point to change curvature. The C^2 path investigated in this thesis allows non-stopping motion of $\mathcal{D}$. It is also subject to a further $\dot{\kappa}_{max}$ constraint to avoid turns that exceed the rate of change of curvature constraint.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT	21. ABSTRACT SECURITY CLASSIFICATION
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS	UNCLASSIFIED
22a. NAME OF RESPONSIBLE INDIVIDUAL Yutaka Kanayama	22b. TELEPHONE *(Include Area Code)* (408) 646-2095 / 22c. OFFICE SYMBOL CSKa

19. ABSTRACT *Continued:*

A class of smooth curves called cubic spirals are adopted for planning C^2 paths. Properties of the cubic spiral are examined in detail. A framework of layered motion planning approach is proposed to divide and conquer the motion planning problem. A novel sensor-oriented method is presented. It plans a spine net which facilitates $\mathcal{D}$ carry out deviation correction using sonar sensors while following a motion path.

Approved for public release; distribution is unlimited.

MOTION PLANNING FOR RIGID BODY ROBOTS

by

Liek Foo Tan
Ministry of Defense, Singapore
B.S., National University of Singapore, 1983

Submitted in partial fulfillment of the
requirements for the degrees of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL

June, 1992

C.1

ABSTRACT

Given a non-holonomic disc robot $\mathcal{D}$, its motion constraints in terms of maximum curvature (κ_{max}) and rate of change of curvature ($\dot{\kappa}_{max}$), a set W of rectilinear polygonal obstacles which assemble an office-like environment, and two configurations S and G in $free(W)$, this thesis investigates the planning of a smooth free path which satisfies the following condition: $\mathcal{D}$ is allowed backing up motions at the end portions of the path, but the middle portion is to be of class C^2 in its entirety. Although the motion planning problem of $\mathcal{D}$ amidst polygonal obstacles has been extensively studied, the paths considered are mostly class C^1 and piecewise C^2 only, and are subject only to the κ_{max} constraint. Typically, such paths consist of straight line segments and circular ars which have curvature discontinuity at the junction points. In order for $\mathcal{D}$ to follow such paths physically, $\mathcal{D}$ has to stop abruptly at each junction point to change curvature. The C^2 path investigated in this thesis allows non-stopping motion of $\mathcal{D}$. It is also subject to a further $\dot{\kappa}_{max}$ constraint to avoid turns that exceed the rate of change of curvature constraint. A class of smooth curves called cubic spirals are adopted for planning C^2 paths. Properties of the cubic spiral are examined in detail. A framework of layered motion planning approach is proposed to divide and conquer the motion planning problem. A novel sensor-oriented method is presented. It plans a spine net which facilitates $\mathcal{D}$ carry out deviation correction using sonar sensors while following a motion path.

TABLE OF CONTENTS

I. INTRODUCTION 1

A. BACKGROUND 1

B. DEFINITIONS 2

C. STATEMENT OF THE PROBLEM 4

1. Motivation and Informal Description 4

2. Formal Statement 5

3. Rationale Behind the Scope of the Problem 5

D. IN COMPARISON WITH THE BASIC MOTION PLANNING PROBLEM 7

E. RELATED WORK 10

F. METHODOLOGY 13

G. ORGANIZATION OF STUDY 15

II. LAYERED MOTION PLANNING APPROACH 16

A. GLOBAL PATH PLANNING LAYER (PREPROCESSING STEP) 17

B. LOCAL SMOOTH PATH PLANNING LAYER (PREPROCESSING STEP) 19

C. END PORTIONS PATH PLANNING LAYER (PER QUERY) . . 20

D. GRAPH SEARCH LAYER (PER QUERY) 20

E. REMARKS 21

III. GLOBAL PATH PLANNING LAYER 22

A. OVERVIEW 22

B. THE VISIBILITY NET 23

1. Introduction 23

2. The Visibility Graph Method 23

3. Data Structure of the Visibility Net 24
4. Time Complexity Analysis of Building the Visibility Net . . 26
5. Critique 26
C. THE NOMINAL CONFIGURATION NET 27
1. Introduction 27
2. The Minimum Gate Method 27
3. Data Structure of the Nominal Configuration Net 31
4. Time Complexity Analysis of Building the Nominal Configuration Net 32
D. THE SPINE NET 33
1. Introduction 33
2. The Freeway Method 33
3. The Sensor-Oriented Method 37
4. Data Structure of the Spine Net 39
5. Time Complexity Analysis of Building the Spine Net 41
IV. CUBIC SPIRALS 43
A. OVERVIEW 43
B. BASIC PROPERTIES OF CUBIC SPIRALS 45
C. CIRCUMSCRIBING CUBIC SPIRALS AND INSCRIBING CIRCLES 51
D. CUBIC SPIRALS SUBJECT TO κ_{max} AND $\dot{\kappa}_{max}$ CONSTRAINTS 54
E. BOUNDING REGIONS OF CUBIC SPIRALS 61
V. LOCAL SMOOTH PATH PLANNING LAYER 64
A. OVERVIEW 64
B. SMOOTHING TECHNIQUE FOR THE VISIBILITY NET 66
1. Shortcomings of the Unconstrained Visibility Net 66
2. Rebuilding the Visibility Graph Subject to κ_{max} Constraint . 66

3. Rebuilding the Visibility Graph Subject to $\dot{\kappa}_{max}$ Constraint . 69
4. The Smoothing Technique 71
5. Time Complexity Analysis 72
C. SMOOTHING TECHNIQUE FOR THE NOMINAL CONFIGURATION NET 75
1. The Smoothing Technique 75
2. Time Complexity Analysis 75
3. Critique 75
D. SMOOTHING TECHNIQUE FOR THE SPINE NET 78
1. Background 78
2. The Smoothing Technique 78
3. Time Complexity Analysis 82
4. Modifications to the Data Structure 82
VI. END PORTIONS PATH PLANNING LAYER 83
A. OVERVIEW 83
B. DEFINITIONS 85
C. MERGING A CONFIGURATION INTO A DIRECTED LINE SEGMENT USING R_{min}–CIRCULAR–ARCS 88
1. Without Considering Obstacles 88
2. Limiting the Solution Ranges 91
3. Considering Obstacles 93
D. DIVERGING A CONFIGURATION OUT OF A DIRECTED LINE SEGMENT USING R_{min}–CIRCULAR–ARCS 97
E. MANEUVERING BETWEEN A CONFIGURATION AND A DIRECTED LINE SEGMENT USING CUBIC SPIRALS 98
1. Merging p into e Using Cubic Spirals Subject to κ_{max} Constraint Only 99

2. Merging p into e Using Cubic Spirals Subject to an Additional $\dot{\kappa}_{max}$ Constraint 101
3. Diverging p out of e Using Cubic Spirals Subject to Both κ_{max} and $\dot{\kappa}_{max}$ Constraints 102
F. END PORTIONS PATH PLANNING FOR THE SPINE NET USING CUBIC SPIRALS 103
G. TIME COMPLEXITY ANALYSIS 105
VII. GRAPH SEARCH LAYER 106
A. OVERVIEW 106
B. PROCEDURE GRAPH SEARCH 107
C. TIME COMPLEXITY ANALYSIS 108
VIII. IMPLEMENTATION PLAN 109
A. INTRODUCTION TO YAMABICO-11 ROBOT 109
B. KINEMATIC CONSTRAINTS OF YAMABICO-11 ROBOT . . . 110
C. DESIRABLE MOTION PATH CHARACTERISTICS 112
D. SPECIFIC COST FUNCTION 112
E. ADDITIONAL CONSIDERATIONS FOR IMPLEMENTATION . 113
IX. CONCLUSION 114
LIST OF REFERENCES 117
INITIAL DISTRIBUTION LIST 119

LIST OF FIGURES

1.1 Various Types of Path 4

1.2 Convolution Diagram of the C–Space of $\mathcal{D}$ 9

2.1 Layered Motion Planning Approach 18

3.1 Vertices and Edges of the Visibility Net $\mathcal{M}$ 25

3.2 Minimum Gates 30

3.3 Nominal Configurations 30

3.4 Vertices and Edges in Nominal Configuration Net 31

3.5 Introducing Additional Edges 32

3.6 Construction of Freeway Spines 36

3.7 Intersecting Directed Spines and Directed Turns 36

3.8 Construction of Spines Using Sensor-Oriented Method 40

4.1 Curvature of Cubic Spiral 46

4.2 A Cubic Spiral Curve in Cartesian Coordinates 46

4.3 Symmetric Configurations 46

4.4 Parameters of Cubic Spirals 48

4.5 A C^2 Path Consisting of Two Line Segments and A Cubic Spiral . . . 48

4.6 A Family of Similar Cubic Spirals with the same deflection α 49

4.7 $D(\alpha)$ Plotted Against α in Steps of 1° 49

4.8 A Circle Inscribing a Cubic Spiral 52

4.9 $F_1(\alpha)$ Plotted Against α in Steps of 1° 53

4.10 $F_2(\alpha)$ Plotted Against α in Steps of 1° 57

4.11 $F_3(\alpha)$ Plotted Against α in Steps of 1° 59

4.12 Bounding Region BR_1 of a Cubic Spiral 61
4.13 Bounding Region BR_2 of a Cubic Spiral 62
4.14 Substituting a Cubic Spiral with a Sequence of Sub-Spirals 63

5.1 Problems of Growing Obstacle Corners by $r_{min} > r_0$ 68
5.2 Circumscribing Circular Vertex with Circle of radius r_{min} 69
5.3 $F_4(\alpha)$ Plotted Against α in Steps of 1° 73
5.4 Finding Another Circle with $chord_{beta}$ 73
5.5 Two Very Close Turns of Angles $< \beta$ 74
5.6 Making Room for Two β–Chords 74
5.7 Working Examples of the Nominal Configuration Net Method 76
5.8 Counter Examples of the Nominal Configuration Net Method 77
5.9 Computing r_{max} for a turn 80
5.10 Collision-Free Critical Intervals of a Turn 81
5.11 Constructing BR_2 for a Critical Interval 81

6.1 Relative Positions of Circles Against a Directed Line Segment 85
6.2 Relative Configurations of Circles Against a Directed Line Segment . 85
6.3 Four Types of Maneuvering Arcs Tangential to p 86
6.4 Tracing Out a New Maneuvering Arc from an Extremity 87
6.5 Solution Paths Connecting Two Configurations 88
6.6 Adequacy of Forward Mergings 91
6.7 The Collision-Free Limited Solution Range 93
6.8 Partially Visible e and the New Truncated e 96
6.9 Final Maneuver in the Initial Phase Using Cubic Spiral Turns 100
6.10 The Collision-Free Limited Solution Range subject to $\dot{\kappa}_{max}$ Constraint 102

I. INTRODUCTION

A. BACKGROUND

In the field of Robotics, the ultimate goal is to design autonomous robots that are artificially intelligent. This thesis is an investigation of a humble aspect of this goal: the problem of motion planning which allows an autonomous robot plan its own motion in a known and static two-dimensional environment. Although the research in robot motion planning can be traced back to the late 60's, most of the theoretical breakthroughs and practical understandings of the issue have been achieved only in the last decade, and much of the problem is still outstanding.

The difficulty of motion planning can best be summarized by J. C. Latombe [Ref. 1] as follows:

> *At first glance motion planning looks relatively simple, since humans deal with it with no apparent difficulty in their everyday lives. In fact, as is also the case with perception, the elementary operative intelligence that people use unconsciously to interact with their environment ... turns out to be extremely difficult to duplicate using a computer-controlled robot. It is true that some naive methods can produce apparently impressive results, but the limitations of these methods quickly become obvious. The unaware reader will be surprised by the amount of nontrivial mathematical and algorithmic techniques that are necessary to build a reasonably general and reliable motion planner.*

The level of complexity of the problem of motion planning again depends on how the robot is being modeled and what physical constraints are imposed on it. To date, even in the simplest case of the *basic motion planning problem* (see Section D.), the problem is considered solved only in very special cases; in general, the basic problem is still open.

As can be seen in the sections that follows, the scope of the motion planning problem in this thesis is carefully defined in such a way as to balance the interests

of theoretical research and engineering applicability within the allowable time frame of the study.

B. DEFINITIONS

This section defines a list of terms and concepts used throughout this thesis.

The environment for the motion planning problem of this thesis is a plane on which a global Cartesian coordinate system is defined. A *world* W is a set of $n(\geq 1)$ simple rectilinear polygonal obstacles,

$$W = \{B_1, \ldots, B_n\}$$

where no pair of polygons are intersecting or touching. The *free space* ($free(W)$) is the complement of the union of all the polygons in W ($free(W)$ includes the boundaries of the polygons).

A *path* π with a finite length l in $free(W)$ is a pair $\pi = (l, f)$, where l is a positive real number and f is a continuous and standard function,

$$f : [0, l] \rightarrow free(W)$$

A path is thus by definition *free* (or collision-free) and lies completely in $free(W)$.

A *configuration* is a combination of position and orientation (x, y, θ). If $p = (x, y, \theta)$ is a configuration, "point (p)" stands for the point (x, y). A disc robot $\mathcal{D}$ in $free(W)$ possesses three degree of freedom which are represented by the configuration of its center.

The geometry of the *disc robot* $\mathcal{D}$ is a circle $C = (x, y, r_0)$, where x, y are the coordinates of the center of $\mathcal{D}$ and $r_0(> 0)$ is its radius. The center of $\mathcal{D}$ is taken as the tracking reference point of a given reference path. Only configuration tracking is considered in this paper, in which the center (reference point) of $\mathcal{D}$ stays on the path and the orientation of the center equals the tangent orientation of the path.

Motion of $\mathcal{D}$ is subject to two *kinematic constraints*:

- A finite *curvature* limitation of motion represented by the maximum curvature (κ_{max}) that the vehicle can take.
- A finite *rate of change of curvature* limitation of smooth motion represented by the maximum rate of change of curvature ($\dot{\kappa}_{max}$).[1]

The disc robot $\mathcal{D}$ is able to perform both forward and backing up (reversal) motions. It is unable to rotate about its center because of the kinematic constraints imposed on it.

A path is of class C^n if it is differentiable n times and if its n^{th} derivatives is continuous. A C^1 path has continuity in tangential direction. An example of which is a path consisting of circular arcs and tangential straight line segments. A C^2 path has an additional continuity in curvature κ. An example of which is a path consisting of cubic spirals and tangential straight line segments.

A C^1 path is κ_{max}*−constrained* if $\mathcal{D}$ is able to follow it under configuration tracking subject to κ_{max} constraint. The κ_{max} constraint is typically imposed by the maximum steering wheel angle of the robot.

A C^2 path is $\dot{\kappa}_{max}$*−constrained* if $\mathcal{D}$ is able to follow it under configuration tracking subject to $\dot{\kappa}_{max}$ constraint. The $\dot{\kappa}_{max}$ constraint is typically imposed by the condition that the robot is not supposed to travel below a speed of $v_{min}(> 0)$ when making a turn.

A path is *feasible* if it is piecewise of class C^2 and κ_{max}−constrained, and if every two adjacent C^2 path pieces share a common vertical tangent forming a "cusp" (see Figure 1.1(a) for an example). It is possible for $\mathcal{D}$ to follow a feasible path with both

[1]This limitation is applicable only when we are interested in smooth motion in which the disc robot $\mathcal{D}$ is not supposed to stop when moving along a path. If $\mathcal{D}$ is allowed to stop before making a maneuvering turn, then this limitation does not exist, and $\mathcal{D}$ is able to follow any κ_{max}−constrained path so long as there is tangential continuity anywhere on the path.

forward and backing up motions.

A path is *near smooth* if it is piecewise of class C^2, in its entirety of class C^1 and κ_{max}−constrained (see Figure 1.1(b) for an example). It is possible for $\mathcal{D}$ to follow a near smooth path with only forward motions, but the robot has to stop at every turn where there is a discontinuity in curvature.

A path is *smooth* if it is entirely of class C^2 and is both κ_{max}−constrained and $\dot{\kappa}_{max}$−constrained (see Figure 1.1(c) for an example). It is possible for $\mathcal{D}$ to follow a smooth path with non-stopping forward motions.

The following relationship holds:

$$\text{smooth paths} \subset \text{near smooth paths} \subset \text{feasible paths}$$

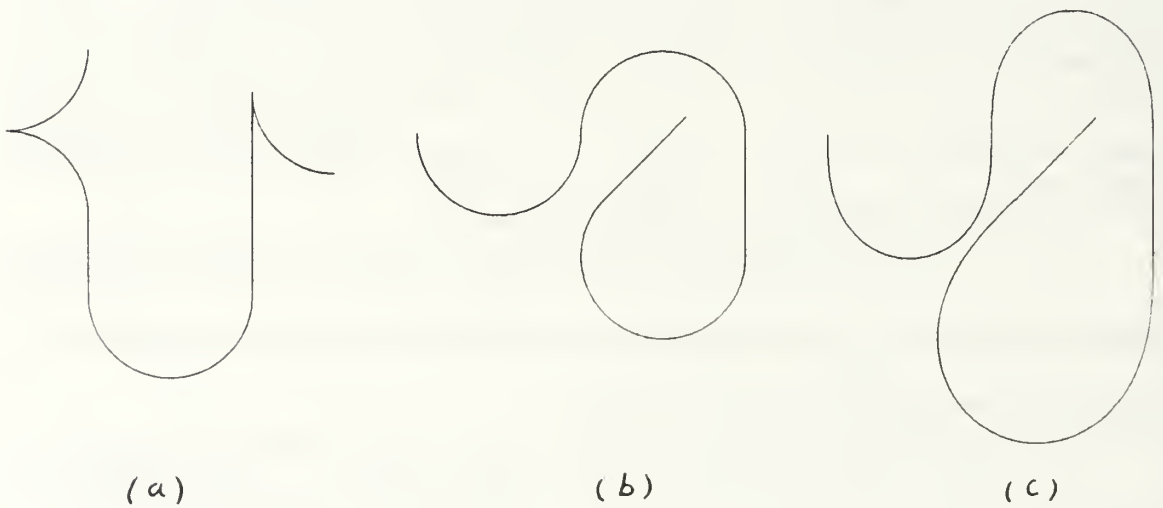


Figure 1.1: Various Types of Path

C. STATEMENT OF THE PROBLEM

1. Motivation and Informal Description

The problem is to plan a motion path for $\mathcal{D}$ in a rectilinear polygonal world which resembles an in-door office environment. The path is to start from

an initial configuration S, end at a final configuration G, and possess the following characteristics:

- Since a typical office environment is expected to have ample leeway for $\mathcal{D}$ to navigate, the path must allow $\mathcal{D}$ to navigate in a non-stopping and forward-moving motion.
- Exceptions are at the initial and the final portions of the path, where depending on the orientations, backing up motions may be required in order for $\mathcal{D}$ to move out of S or into G. This requirement could be well understood by the everyday driving experience of the reader.

2. Formal Statement

Let S and G be two given configurations of the disc robot $\mathcal{D}$ lying completely inside $free(W)$; S is called the initial configuration and G the final configuration. The problem is to find an intermediate S_0 (can be coincident with S) and an intermediate G_0 (can be coincident with G), and plan a path from S to S_0 to G_0 to G satisfying the following requirements:

1. The entire path from S to G must be *feasible.*[2]
2. The middle portion of the path from S_0 to G_0 must be *smooth.*

As a corollary, the initial portion of the path from S to S_0, and the final portion from G_0 to G, must both be *feasible.*

3. Rationale Behind the Scope of the Problem

This subsection explains the rationale behind the definition of the scope of the motion planning problem in this thesis. It is defined with the following con-

[2]Please refer to Section B. for the definition of feasible and smooth paths.

siderations in order to balance the interests of theoretical research and engineering applicability within the allowable time frame of the study:

1. The problem should be manageable

   In order to simplify the treatment of geometric reasoning, the world consists of only two-dimensional rectilinear polygonal obstacles, and the robot is modeled in disc-shaped.

2. The problem should be broad enough in scope

   Non-holonomic and kinematic constraints are imposed on the disc robot so that the study result could apply to a wide range of rigid body robots modeled in disc-shaped. The study result should also be amenable to extensions to general polygonal world and non-disc-shaped robots.

3. The problem should pose research challenge

   The desired characteristics of the planned path, which calls for feasible end portions and smooth middle portion, are issues not well-established in the Robotic field. Yet they are realistic requirements of paths to be planned in an office-like environment, and are worth the research effort for investigation.

4. The problem should be practical

   There is a research robot Yamabico-11 under continuous development by students and faculty at the Naval Postgraduate School. Its target operating environment is an in-door office environment with mostly rectilinear type of constructions and obstacles. The general findings of this thesis should be able to be readily tailored for implementation on the Yamabico-11 robot within the context of its target operating environment.

D. IN COMPARISON WITH THE BASIC MOTION PLANNING PROBLEM

In his book "Robot Motion Planning"[Ref. 1], J. C. Latombe defines the **basic motion planning problem** (or the *classical path planning problem*) as follows:

> *Let $\mathcal{A}$ be a single rigid object (the robot) moving in a Euclidean space $\mathcal{W}$, called* workspace, *represented as $\mathbf{R}^N$, with $N = 2$ or 3.*
>
> *Let $\mathcal{B}_1, \ldots, \mathcal{B}_q$ be fixed rigid objects distributed in $\mathcal{W}$. The $\mathcal{B}_i$'s in $\mathcal{W}$ are accurately known. Assume further that no kinematic constraints limit the motions of $\mathcal{A}$ (we say that $\mathcal{A}$ is a* free-flying object*).*
>
> *The problem is: Given an initial position and orientation and a goal position and orientation of $\mathcal{A}$ in $\mathcal{W}$, generate a* path *τ specifying a continuous sequence of positions and orientations of $\mathcal{A}$ avoiding contact with the $\mathcal{B}_i$'s, starting at the initial position and orientation, and terminating at the goal position and orientation. Report failure if no such path exists.*

The scope of this thesis is a simplification of this basic motion planning problem in some aspects, and an extension in other aspects:

- *Simplifications*:

  1. The robot $\mathcal{D}$ is modeled as a disc robot. Because of its geometry simplicity, when $\mathcal{D}$ is represented as a point in its configuration space (*C–space*)[Ref. 2], its C–space is only two-dimensional. The obstacles in the C–space (called the C–obstacles) are those obstacles in $\mathcal{W}$ grown by a bumper of depth r_0 which is the radius of $\mathcal{D}$. The resulting diagram of the C–obstacles is called the *convolution diagram*[Ref. 3] (see Figure 1.2). In contrast to disc robots, the C–space of non-disc-shaped rigid body robot is always three-dimensional which makes motion planning a more difficult task.

  2. The world W is only of $\mathbf{R}^2$, and consists only of rectilinear polygonal obstacles.

- *Extensions*:

1. The robot $\mathcal{D}$ is non-holonomic and unable to rotate about its center. Thus, the motion path is required to have tangential continuity. This extension of the problem has been extensively studied.

2. The robot $\mathcal{D}$ is subject to the kinematic constraint of κ_{max}. An example of such constraint is the limited steering range of a vehicle. This extension of the problem has been extensively studied in [Ref. 1: Chapter 9] [Ref. 4][Ref. 5] [Ref. 6][Ref. 7] [Ref. 8][Ref. 9].

3. Except at the initial and final portions of the motion path where backing up motions are allowed, the robot $\mathcal{D}$ is only allowed to move forward in a non-stopping manner for the middle portion of the path in between. An example of such a situation is moving a car from one place of a city to another without having to stop for traffic lights or pedestrians. The forward-motion part of this extension for the middle portion of the path has been studied in [Ref. 1: Chapter 9] [Ref. 6][Ref. 7] [Ref. 8][Ref. 9]. However, there has not been much investigation by researchers on the non-stopping-motion part of this extension which calls for additional continuity in curvature.

4. The robot $\mathcal{D}$ is subject to an additional kinematic constraint of $\dot{\kappa}_{max}$ depending on its traveling speed. An example of such constraint is that a non-stopping vehicle when making a turn is subject to a maximum rate of change of steering angle. This extension of the problem, to the best of the authors' knowledge, has not been investigated before.

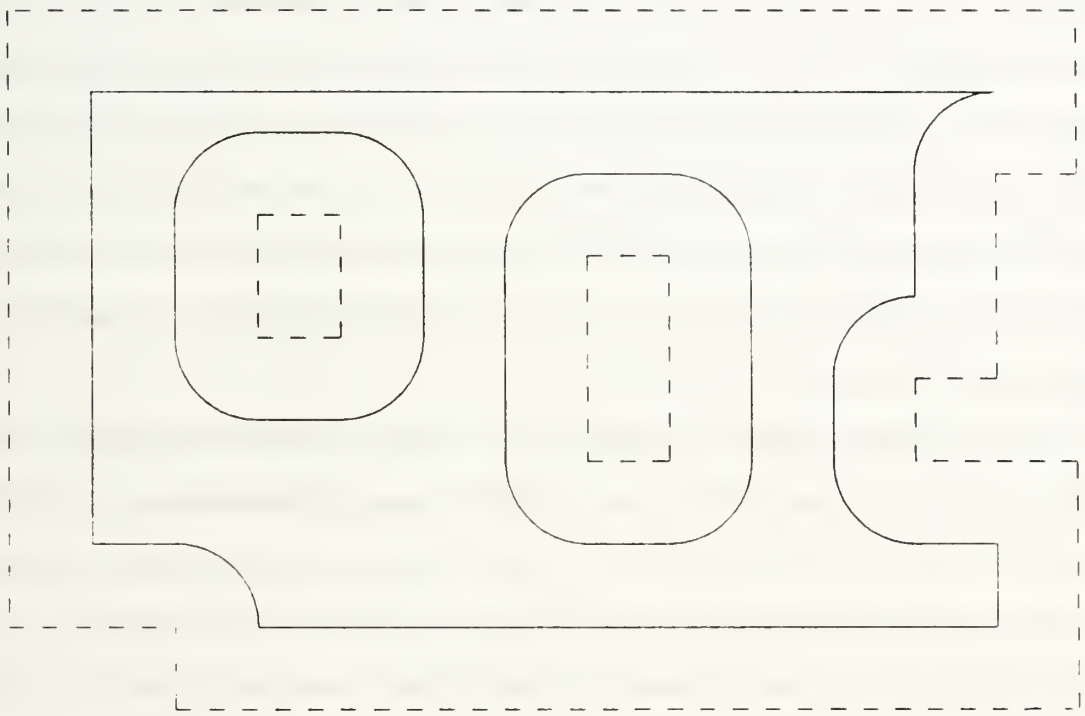


Figure 1.2: Convolution Diagram of the C–Space of $\mathcal{D}$

E. RELATED WORK

In the basic motion planning problem, planning a collision-free path for an unconstrained disc robot $\mathcal{D}$ in the free space of $\mathcal{W}$ ($\subset \mathbf{R}^2$) is equivalent to planning a connected path for a reduced point-$\mathcal{D}$ in the free space of C-space. By all accounts, the basic motion planning problem for unconstrained $\mathcal{D}$ (not irregularly-shaped rigid body robot in general) can be considered solved, particularly under the cost metrics of "shortest distance [Ref. 3]" or "maximum safety clearance [Ref. 10]". Note that in the basic motion planning problem, since $\mathcal{D}$ is not subject to any kinematic constraints, it can stop anywhere and change its orientation at the same spot by rotating about its center. Therefore, the solution path can be in arbitrary shape that requires no tangential continuity. The initial and final orientations are irrelevant. Backing up motions are not necessary at all.

The basic motion planning problem for $\mathcal{D}$ becomes more interesting and challenging when it is extended by imposing real-life kinematic constraints on the robot. Interests in the extension problem have been intensified after Laumond published his results in 1986 [Ref. 4]. In his paper, Laumond proved that in the case of a point robot with κ_{max} constraint, for every two configurations connected by a collision-free path in $free(W) \subset \mathbf{R}^2$, there exists a feasible path between them which satisfies the κ_{max} constraint. His method breaks down the planning problem into two phases. In a first phase, the problem is solved as a basic motion planning problem in that an (arbitrary) collision-free path is found ignoring the orientations of S and G. In a second phase, this path is transformed into a topologically equivalent collision-free path using only R_{min}-arcs and tangent line segments. Unfortunately, the number of maneuvers (reversals) introduced in the feasible path constructed by his proof is not bounded, and hence the computation complexity is difficult to analyze. Also, a path involving reversals throughout its course are not smooth and are required only in

very constrained environment. This algorithm is generally regarded as having high theoretical value but little practical use.

Another feasible path planning algorithm presented by Vasseur et al. [Ref. 5] takes a different approach. Reversals are still allowed throughout the path for a non-holonomic car-like vehicle. However, for path planning, they break up $free(W)$ into convex cells, and calculate intermediate configurations at the border of every adjacent pair of cells. They then maneuver the vehicle from S to G via all the in-between intermediate configurations. Since all the intermediate configurations of a convex cell are visible to one another, a simple procedure is devised to connect them using only R_{min}–arcs and tangent lines (probably after some numbers of reversals). This method seems complete but the authors did not claim so explicitly in their paper. The computation bound was also not discussed because the number of reversals involved may be unbounded. The simulation results showed that the solution path involves a lot of unnecessary reversals even in a spacious environment, and looks unnatural and awkward.

On the other hand, there are difficulties in planning near smooth paths for $\mathcal{D}$ without reversals (smooth path planning has not been well investigated by researchers). If reversal maneuvers are disallowed in a path all together, it has been shown (see [Ref. 1] for a good introduction) that a forward moving rigid body robot (and hence $\mathcal{D}$) with κ_{max} constraint is not fully controllable.[3] At present, it is still an open question regarding the lower bound on the problem of finding a near smooth (κ_{max}–constrained) path even for just *point robots* [Ref. 11]. One of the best results for point robots so far is credited to G. Wilfong and S. Fortune. In [Ref. 6], they gave an exponential complexity decision procedure for this problem, although their algorithm does not find the actual path.

[3]A robot is said to be fully controllable if and only if, for any distribution of obstacles in the workspace, if there exists a free path between any two configurations S and G, then there also exists a feasible path between these two configurations [Ref. 1 : Chapter 9].

Because of these difficulties, many researchers have come up with incomplete algorithms to plan near smooth paths for forward-moving rigid body robots (and hence $\mathcal{D}$) subject to κ_{max} constraint. They normally start with a global planning that produces a network of "corridors" [Ref. 8], "lanes" [Ref. 7] or "spines"[Ref. 9] that are extracted from $free(W)$. Local planning techniques are then used to generate circular turns with curvature smaller that κ_{max} for transferring the robot from a corridor (or lane or spine) to another at a junction of the network. Invariably, all of them avoided the end portions problem by conveniently assuming that S and G lie on the spines and point to one of the two directions of a spine. Also, the solution paths are only near smooth, which consist of straight line segments and circular arcs. They are piecewise of class C^2, but are of class C^1 in their entirety. At the junction of a line segment and a circular arc, the curvature is not continuous. Thus, a non-holonomic robot has to stop at each junction abruptly in order to follow the paths. This problem normally arises from the choice of circular arcs for junction turns, but not because of the constrained environment.

From the author's point of view, none of these surveyed algorithms is suitable for planning a motion path for $\mathcal{D}$ in a rectilinear polygonal world resembling an in-door office environment. That is, none of them is a satisfactory answer to the problem statement set forth in this thesis. Of these existing results, the author opines that the approach of the incomplete algorithms which plan near smooth paths for forward-moving robots subject to κ_{max} constraint provides a most useful framework for this thesis to work with. Based on such a framework and compilation of other related work, this thesis further investigates the following open issues:

1. Layered Motion Planning Approach

   Based on the initial idea of breaking down the motion planning problem into phases, a formalized layered motion planning approach is fully explored.

2. Sensor-Oriented Global Path Planning Method

   Of the existing global path planning methods, none is suitable for robots which desire to follow parallel walls for position and orientation correction using sonar returns. A novel sensor-oriented method is proposed in this thesis.

3. Local Smooth Path Planning

   To plan for smooth paths instead of near smooth ones, a class of curves called cubic spirals [Ref. 12] instead of circular arcs are used to generate turns. The properties of cubic spirals under the kinematic constraints are thoroughly investigated.

4. End Portions Path Planning

   To the best of the authors' knowledge, this problem has not been investigated before; Related studies in this area as discussed above are not directly relevant and applicable to this paper either.

F. METHODOLOGY

One contribution of this thesis is the laying down of a formal framework of layered motion planning approach to tackle the full problem of motion planning. Preliminary attempts of such an approach can be found in [Ref. 6][Ref. 7] [Ref. 8][Ref. 9]. In this thesis, the task of motion planning is formally broken down into the following four layers:

1. Global path planning layer (preprocessing step).

2. Local smooth path planning layer (preprocessing step).

3. End portions path planning layer (per query).

4. Graph search layer (per query).

In [Ref. 1: Chapter 4], Latombe groups a large number of methods for solving the basic motion planning problem into the following three broad categories and sub-categories:

1. Roadmap Approach
    (a) Visibility Graphs
    (b) Voronoi Diagrams
    (c) Freeway Nets
    (d) Silhouettes
2. Cell Decomposition Approach
    (a) Exact Methods
    (b) Approximate Methods
3. Potential Field Approach

This thesis adopts only the roadmap approach for the global path planning layer. For the local smooth path planning layer, this thesis adopts a class of smooth curves called cubic spirals [Ref. 12] to design paths of curvature continuity. The end portions path planning layer represents an original effort of this thesis. The graph search layer makes a Dijkstra's search of the roadmap to find an optimal solution for the motion planning problem.

G. ORGANIZATION OF STUDY

The layered motion planning approach, which divides the problem statement of this thesis into four layers and conquers them separately, is explained in Chapter II. This should give the reader a good overview of the approach before each layer is examined in detail. Chapter III explores a number of roadmap methods that can be employed in the global path planning layer. A novel sensor-oriented roadmap method is introduced. Chapter IV is devoted to a detailed investigation on the properties of a class of C^2 curves called cubic spirals which will be used in the subsequent chapters for planning smooth turns. Chapter V investigates the techniques of generating smooth cubic spiral turns for the global roadmaps in the local smooth path planning layer. Chapter VI proposes a method for planning the end portions of paths in smooth roadmaps. Chapter VII uses Dijkstra's Search in the graph search layer to select an optimal solution path out of a number of candidates based on some cost functions. At this point, an answer to the problem statement of this thesis will have been offered. Chapter VIII offers some thoughts on how the results of this thesis could be tailored for implementation on the Yamabico-11 robot. Finally, conclusions and avenues for future research are presented in Chapter IX.

II. LAYERED MOTION PLANNING APPROACH

As discussed in Chapter I, the task of motion planning is a formidable undertaking. Even in the scaled down problem of motion planning for a forward-moving point robot subject only to the κ_{max} constraint and the C^1 requirement, there is still no known exact algorithm for a solution. Many researchers thus resort to incomplete but practical algorithms by breaking similar scaled down problems into two layers: global path planning and local path planning.

In this thesis, the problem statement is considerably more involved. It could be conceived that any attempt for an exact and complete solution would be unmanageable within the allowable resources and schedule of this study. Therefore, the same line of thinking by attacking the full problem statement via layered motion planning approach is adopted. The task of motion planning is formally divided into the following four layers:

1. Global path planning layer (preprocessing step).

2. Local smooth path planning layer (preprocessing step).

3. End portions path planning layer (per query).

4. Graph search layer (per query).

Besides dividing and conquering the motion planning problem, this layered motion planning approach offers another advantage of modularity. The same class of roadmaps generated from the global path planning layer can be smoothed out into C^2 roadmaps using a common technique of the local smooth path planning layer.

A common end portions path planning method can be applied to the same class of smooth C^2 roadmaps. Finally, a common graph search technique can be applied to select an optimal solution path out of a number of candidates based on different cost functions defined for each type of roadmap. Figure 2.1 depicts such a modularity concept.

A. GLOBAL PATH PLANNING LAYER (PREPROCESSING STEP)

Given W, the goal of this preprocessing step is to capture the connectivity of $free(W)$ in the form of a network of one-dimensional curves lying in $free(W)$. This network of one-dimensional curves is called a *roadmap*. Various methods based on different principles have been proposed, producing various sorts of roadmaps, called "visibility graphs", "freeway nets", "Voronoi diagrams", and "silhouettes"[Ref. 1: Ch. 4].

In this thesis, roadmaps are classified according to their geometric properties, not the methods from which they are generated. A roadmap is represented as $\mathcal{M} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the collection of all vertices, and $\mathcal{E}$ the collection of all edges. Roadmaps of the same class will have similar basic data structure representation for $\mathcal{M}$. At present, three kinds of roadmaps have been classified in the global path planning layer:

1. Visibility Net

   This is a graph consists of circular arcs and common tangent line segments. The vertices are the tangent points. The edges are the directed circular arcs and the directed tangent line segments. An example of the method that generates this kind of roadmap is the well-known visibility graph method.

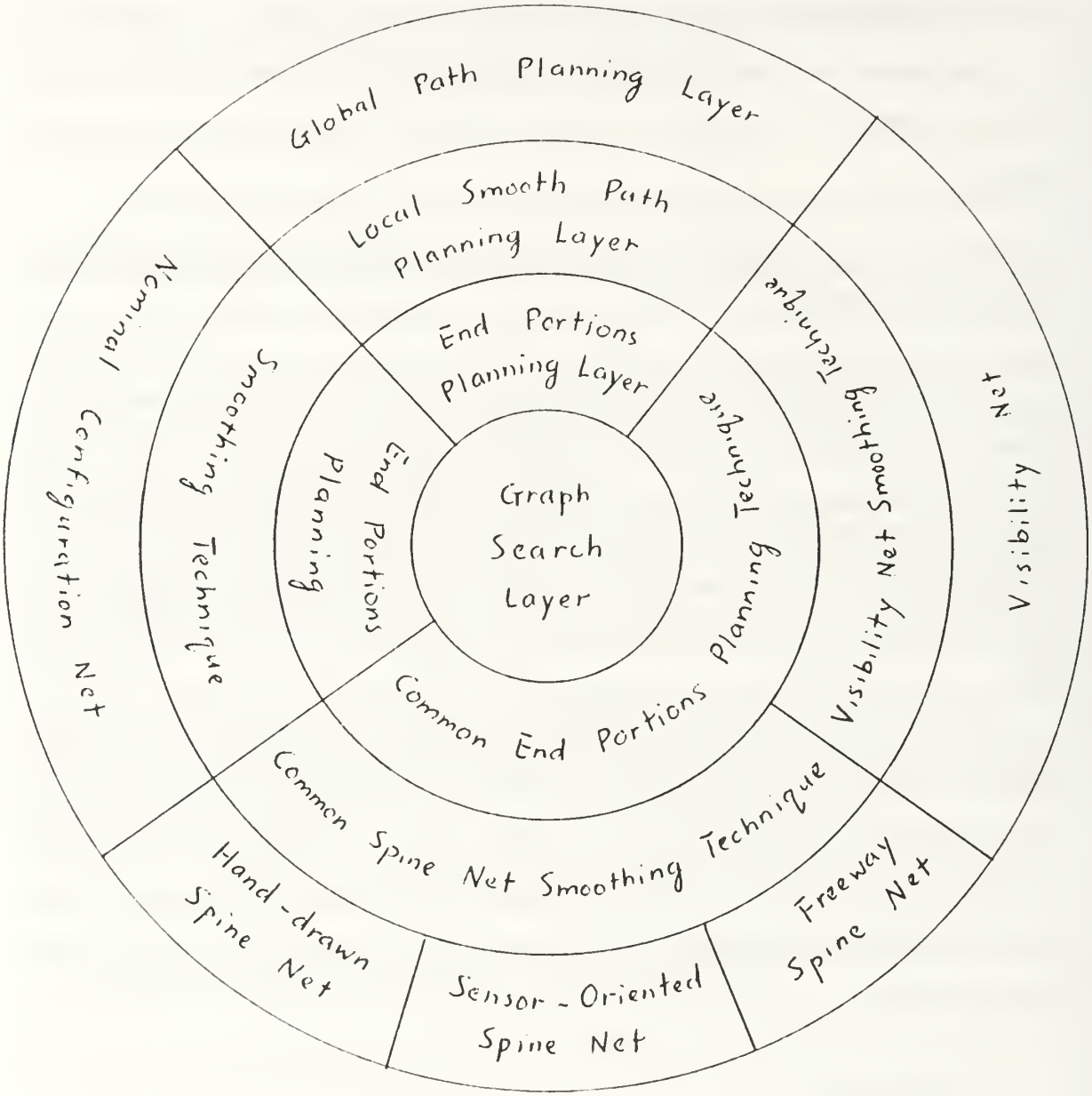


Figure 2.1: Layered Motion Planning Approach

2. Nominal Configuration Net

   This is a graph consists of logically interconnecting nominal configurations. The vertices are the nominal configurations. The edges are the logical connections between some pairs of the vertices. An example of the method that generates this kind of roadmap is a novel minimum gate method proposed in this thesis.

3. Spine Net

   This is a graph consists of only straight line segments called spines. The vertices are the directed turns at the intersection junctions between two directed spines. The edges are the sections of directed spines connecting two directed turns. An example of the method that generates this kind of roadmap is the well-known freeway net method. A novel sensor-oriented method is also proposed in this thesis.

B. LOCAL SMOOTH PATH PLANNING LAYER (PREPROCESSING STEP)

Given a roadmap $\mathcal{M}$ of $free(W)$, the goal of this preprocessing step is to generate collision-free cubic spirals satisfying the κ_{max} and $\dot{\kappa}_{max}$ constraints for all turns in the roadmap. Since not all turns are possible to be realised by constrained cubic spirals, those impossible vertices and edges are removed from $\mathcal{M}$. The resulting smoothed out roadmap is called a *smooth roadmap.*

Note that since straight line segments are just special cases of cubic spirals, the smooth roadmap is said to consist of only cubic spiral segments. *Any path in the smooth roadmap is a smooth path of class* C^2.

C. END PORTIONS PATH PLANNING LAYER (PER QUERY)

A query is a request to plan a smooth path for an initial configuration S and a final configuration G in $free(W)$. In this layer, the problem of motion planning is reduced to finding some candidates of intermediate S_0 (can be coincident with S) and intermediate G_0 (can be coincident with G) such that:

1. It is possible to plan a feasible path from S to S_0.
2. It is possible to merge S_0 into the smooth $\mathcal{M}$ using only cubic spirals.
3. It is possible to diverge from the smooth $\mathcal{M}$ into G_0 using only cubic spirals.
4. It is possible to plan a feasible path from G_0 to G.

At the end of this layer, a number of candidate intermediate configurations of S_0 and G_0 would have been found.

D. GRAPH SEARCH LAYER (PER QUERY)

In this layer, a Dijkstra's search is made for each pair of the candidate S_0 and G_0 on the smooth $\mathcal{M}$. A number of candidate solution paths which meet the path characteristics as set forth in the problem statement will be produced. An optimal solution path from S to S_0 to G_0 to G will be selected out of these candidate solution paths based on some cost functions. Factors that can be considered in a cost function include:

- path length
- path smoothness
- safety

- number of turns
- number of reversals
- wall following distance

Actual definition of a cost function is not important in theoretical studies. However, in actual applications, specific cost functions must be defined for each specific case.

E. REMARKS

In a cramped and cluttered environment with narrow passages, this layered motion planning approach may fail to find a solution even if one exists. However, this incompleteness should not be a serious problem because the target operating environment for the disc robot $\mathcal{D}$ consists of only rectilinear polygonal obstacles which represents realistics office environment with ample leeway for maneuvering. Where this approach fails to find a solution path, the environment would have been too constrained even in the eyes of human beings. More importantly, this approach allows fast computation and satisfactory results of smooth paths, and makes it possible for onboard implementation in the computer system of $\mathcal{D}$.

III. GLOBAL PATH PLANNING LAYER

A. OVERVIEW

The Global path planning layer is the first layer in the layered motion planning approach proposed in this thesis. The goal of this layer is to capture the connectivity of $free(W)$ in the form of a network of one-dimensional curves lying in $free(W)$. Thus reducing the two-dimensional motion planning problem to one that is basically a one-dimensional graph search problem. This network of one-dimensional curves is called a *roadmap*. Given W, this layer is a preprocessing step that needs to be carried out only once to serve all subsequent on-line queries of motion planning.

Various methods based on different principles have been proposed, producing various sorts of roadmaps. In this thesis, roadmaps are classified according to their geometric properties, not the methods from which they are generated. A roadmap is represented as $\mathcal{M} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the collection of all vertices, and $\mathcal{E}$ the collection of all edges. Roadmaps in the same class will have similar basic data structure representation for $\mathcal{M}$. At present, three kinds of roadmaps have been classified in the global path planning layer, namely, the visibility net, the nominal configuration net and the spine net.

In the following sections, each class of roadmaps will be discussed in detail. Discussions will focus on the various methods that can be used to generate the same class of roadmap, time complexity analyses, and a sketch of the basic data structure representation for the roadmap $\mathcal{M}$.

B. THE VISIBILITY NET

1. Introduction

This is a graph consists of circular arcs and common tangent line segments. The vertices in the visibility net are all the tangent points where the tangent line segments touch the circular arcs. The edges in the net are the directed circular arcs and the directed tangent line segments. The only method that generates this kind of roadmap is the well-known visibility graph method.

2. The Visibility Graph Method

a. Background

Planning the shortest path for a disc robot $\mathcal{D}$ using visibility graph method for the basic motion planning problem is a well-known problem. Chew [Ref. 3] presented a complete algorithm solving the problem in $O(n^2 \log n)$ time where n is the number of edges that make up the polygonal obstacles in W. However, when $\mathcal{D}$ is subject to κ_{max} constraint, finding shortest *near smooth paths* amongst obstacles is still an open question. Even if the "shortest-path" requirement is dropped, no one has yet invented a polynomial time complete procedure to find near smooth paths for $\mathcal{D}$. *Let alone the* $\dot{\kappa}_{max}$ *constraint for smooth paths.*

Nonetheless, the visibility graph method can be employed in the global path planning layer to come up with a roadmap $\mathcal{M}$ (visibility net) that lies completely in $free(W)$. Subsequently, in the local smooth path planning layer, attempts could be made to modify $\mathcal{M}$ into smooth $\mathcal{M}$ by replacing the circular arcs in $\mathcal{M}$ by cubic spirals.

b. Details of the Visibility Graph Method

The following procedure which builds the visibility net $\mathcal{M} = (\mathcal{V}, \mathcal{E})$ for $\mathcal{D}$ in the basic motion planning problem is extracted from the work of Chew[Ref. 3].

Procedure Build Visibility Net:

1. **Step 1** Construct the convolution diagram

   First construct the convolution diagram of the configuration space (C–space) for $\mathcal{D}$ so that $\mathcal{D}$ can be represented as a point. The obstacles in the C–space (called the C–obstacles) are those obstacles in W grown by a bumper of depth r_0 which is the radius of $\mathcal{D}$. We call the resulting diagram of the C–obstacles the convolution diagram (see Figure 1.2). The convolution diagram can be constructed in $O(n^2 \log n)$ time using simple ray sweep technique, or in $O(n \log n)$ time using the more complex Voronoi diagram as a preprocessing step.

2. **Step 2** Build the visibility net

   For each circular vertex in the convolution diagram, apply the ray sweep technique on the tangent ray emanating from it to determine all other circular vertices that are visible. This step takes a total of $O(n^2 \log n)$ to build the visibility net.

3. Data Structure of the Visibility Net

Suppose there are n collision-free circular vertices in the convolution diagram. Between a pair of collision-free circular vertices, there are at most eight common directed tangent edges and eight directed tangent contacts (a directed tangent contact is the directed point where a tangent edge touches a circular vertex). Therefore, there are at most $O(n^2)$ directed tangent edges and $O(n^2)$ directed tangent contacts in the convolution diagram.

The vertices of the visibility net $\mathcal{M}$ are the "visible directed tangent contacts" lying on the exterior portion of the circular vertices.[1] The edges are the "visible directed tangent edges" connecting two vertices from different circular vertices, and the "visible directed circular arcs" connecting two vertices on the same circular vertex. See Figure 3.1 for an illustration.

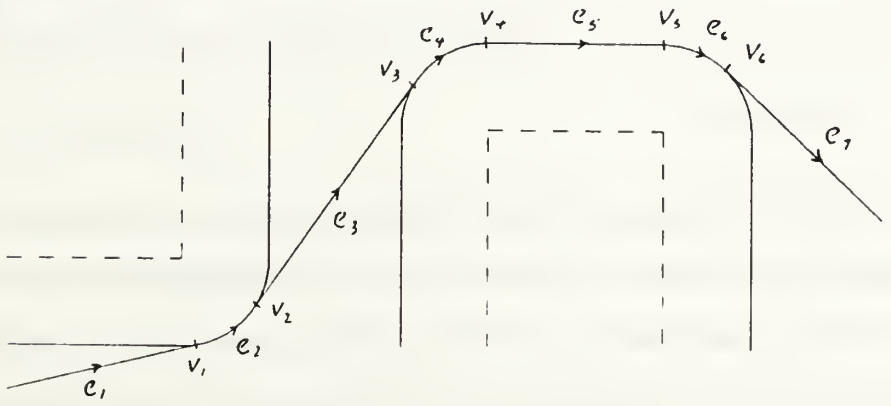


Figure 3.1: Vertices and Edges of the Visibility Net $\mathcal{M}$

[1] A directed tangent contact is "invisible" if it lies on the interior potion of the circular vertex which is inside a C–obstacle.

4. Time Complexity Analysis of Building the Visibility Net

Assuming there are n obstacle edges in W. Building the convolution diagram takes $O(n^2 \log n)$ time using simple ray sweep technique, or $O(n \log n)$ time using Voronoi diagram as a preprocessing step. Constructing the visibility net takes $O(n^2 \log n)$ time.

Conclusion Building the visibility net in the global path planning layer takes $O(n^2 \log n)$ time.

5. Critique

It will be shown in Chapter V (local smooth path planning layer) that the visibility net $\mathcal{M}$ built from *Procedure Build Visibility Net* of this section is not amenable to smoothing techniques. Rather, this procedure must be modified to take into account kinematic constraints faced by $\mathcal{D}$. Because the modification involves understanding of the kinematic constraints imposed on the cubic spirals, the discussion of this issue will be deferred until Chapter V after the properties of the cubic spirals have been thoroughly investigated in Chapter IV. The fact that the kinematic constraints of $\mathcal{D}$ must be taken into account in the global path planning layer makes the visibility graph method inferior to other methods.

C. THE NOMINAL CONFIGURATION NET

1. Introduction

This is a graph consists of logically interconnecting nominal configurations. The vertices are all the nominal configurations. The edges are logical connections between some pairs of the vertices. A novel minimum gate method is proposed in this thesis to generate this kind of roadmap.

2. The Minimum Gate Method

a. Background

The minimum gate method is a novel concept that stems from the concept of Voronoi diagram. The intuition behind this new method is the observation that *minimum gates* (or bottlenecks) of a path occur at those points where the safety clearance is minimal. If the disc robot $\mathcal{D}$ cannot make it through a minimum gate, the whole path is infeasible. Therefore, it may be more efficient and intuitively appealing to focus the planning of a global roadmap around these minimum gates.

However, as stated in the Introduction Chapter, *some naive methods can produce apparently impressive results, but the limitations of these methods quickly become obvious.* This novel method is a good demonstration of this statement. It will be shown later that while this method is elegant and efficient in some situations, it quickly fails in some non-trivial counter-examples.

In the following discussion of this novel method, formal definitions are first given for the minimum gate and the nominal configuration, followed by a discussion of how to extract them from W, and how to construct a nominal configuration net out of them.

b. Definitions

The following notations are adopted for the subsequent definitions:

- Let $vor(W)$ denote the Voronoi diagram of W. It consists of straight and parabolic arcs.
- Denote a straight arc of $vor(W)$ by e, and a parabolic arc by c.
- Denote the two end points of a Voronoi arc by $t_1(e)$, $t_2(e)$, $t_1(c)$ and $t_2(c)$, etc.
- Denote by $p(e)$ and $p(c)$ any point on e and c respectively, and we say $p(e) \in e$ and $p(c) \in c$.
- Denote by $w[p(e)]$ and $w[p(c)]$ the clearance of $p(e)$ or $p(c)$ from the nearest obstacle in W.

A *spine* is an $e \in vor(W)$, where $\forall p_1(e), p_2(e) \in e, w[p_1(e)] = w[p_2(e)]$. See Figure 3.2(a).

A *skew* is an $e \in vor(W)$, where $\exists p_1(e), p_2(e) \in e$, such that $w[p_1(e)] \neq w[p_2(e)]$. See Figure 3.2(b).

A *twisted pair* is a pair of consecutive $c_1, c_2 \in vor(W)$, where $t_2(c_1) = t_1(c_2)$, and $w[t_1(c_1)] > w[t_2(c_1)]$ and $w[t_1(c_2)] < w[t_2(c_2)]$. A twisted pair occurs at two opposite corners lying on opposite sides of a line. See Figure 3.2(c).

A *local minimum point* is either the point on a skew that is closest to the two bisected corners, or the minimum point on a twisted pair (which is also the closest point bisecting two corners).

A *minimum gate* is either an end point of a spine, or a local minimum point. More specifically, let $p(\mathcal{V})$ denote the set of minimum gates. Then the minimum gates are the following points on $vor(W)$:

1. For an $e \in vor(W)$, if $\forall p_1(e), p_2(e) \in e, w[p_1(e)] = w[p_2(e)]$, then $t_1(e), t_2(e) \in p(\mathcal{V})$. See Figure 3.2(a).

2. For an $e \in vor(W)$, if $\exists p(e) \in e$, such that $\forall p_3(e) \in e, p_3(e) \neq p(e) \Leftrightarrow w[p(e)] < w[p_3(e)]$, then $p(e) \in p(\mathcal{V})$. See Figure 3.2(b).

3. For every two consecutive c_1 and c_2 with $t_2(c_1) = t_1(c_2)$, if $w[t_1(c_1)] > w[t_2(c_1)]$ and $w[t_1(c_2)] < w[t_2(c_2)]$, then $t_2(c_1) \in p(\mathcal{V})$. See Figure 3.2(c).

Each minimum gate can be assigned two tangential orientations, θ and $\theta + 180°$. That is, each minimum gate defines two **nominal configurations**. See Figure 3.3. The set of *nominal configurations* is denoted by $\mathcal{V}$, which is also the set of vertices of the nominal configuration net $\mathcal{M}$.

c. Details of the Minimum Gate Method

The nominal configuration net $\mathcal{M}$ can be constructed as follows:

Procedure Build Nominal Configuration Net:

1. **Step 1** Extract the Nominal Configurations

   From the definitions of minimum gates and nominal configurations, we can see that it is straight forward to extract nominal configurations from $vor(W)$. See Figure 3.3.

2. **Step 2** Build the Nominal Configurations Net $\mathcal{M}$

   Let $\mathcal{M} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the set of nominal configurations. $\mathcal{E}$ can be constructed as follows:

   (a) Two minimum gates are connected if there is a path in $vor(W)$ connecting them. From two connected minimum gates, we can generate two edges

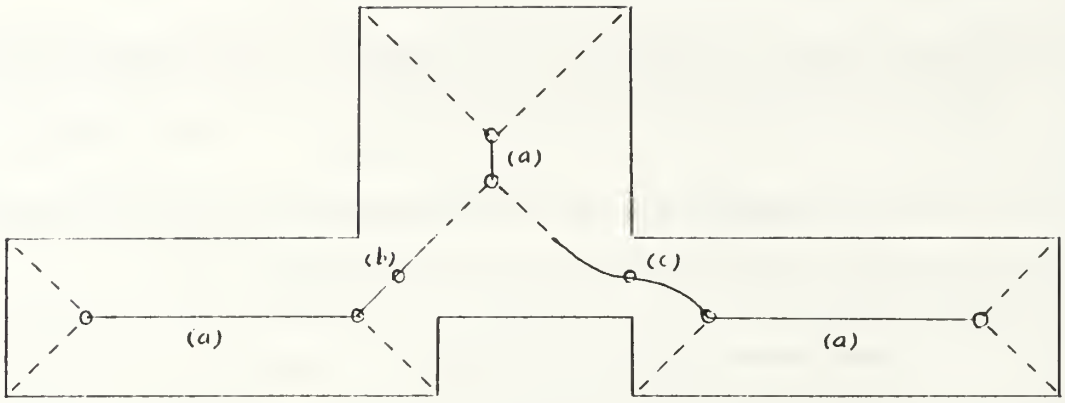


Figure 3.2: Minimum Gates

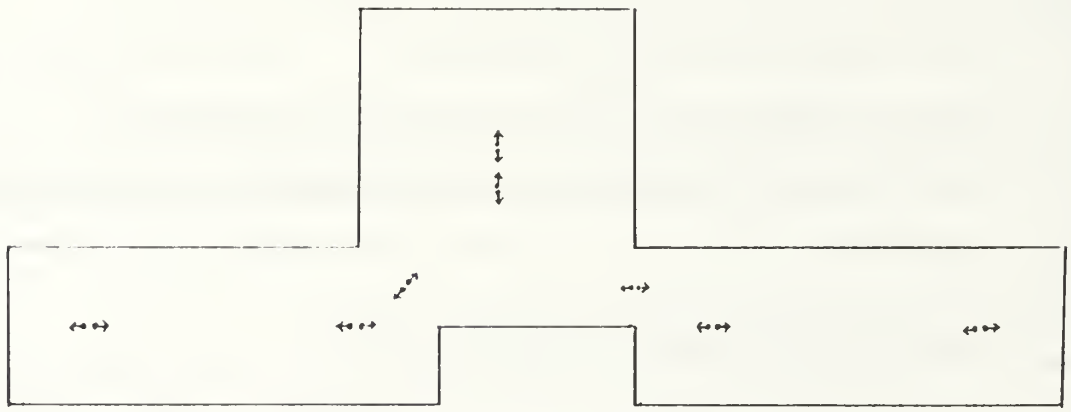


Figure 3.3: Nominal Configurations

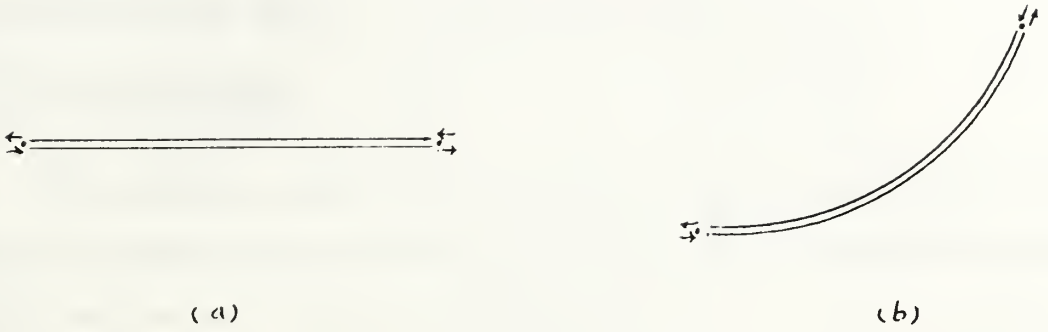


Figure 3.4: Vertices and Edges in Nominal Configuration Net

(directed paths) connecting two vertices (nominal configurations). See Figure 3.4(a) and 3.4(b)

(b) Consider the case where a minimum gate, say v_1, has a degree of three (see v_1 of Figure 3.5 for an example).[2] Supposing v_1 is connected to v_2, v_3 and v_4. If v_3 and v_4 are not connected to v_1 by spines, it is necessary to introduce an additional pseudo connection between v_3 and v_4. As a result, we introduce two additional edges in $\mathcal{E}$. This step seems strange, but is meant to resolve an inherent difficulty of this method as illustrated in Figure 3.5.

3. Data Structure of the Nominal Configuration Net

The vertices of the nominal configuration net $\mathcal{M}$ are the nominal configurations. The edges are the logical connections between nominal configurations. Note that in contrast to the visibility graph method and the spine net method to be discussed shortly, the edges in the nominal configuration net are not physical collision-free paths, but are just logical concepts. The construction of the physical collision-free paths are left to the next layer, the local smooth path planning layer.

[2]There is a common assumed property of Voronoi diagram that the maximum degree of a vertex is three.

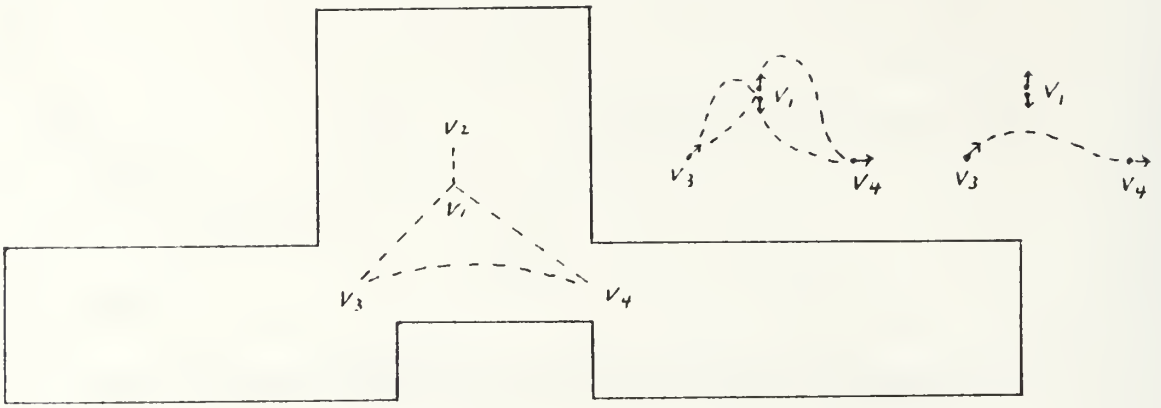


Figure 3.5: Introducing Additional Edges

4. Time Complexity Analysis of Building the Nominal Configuration Net

Assuming there are n obstacle edges in W. Building the Voronoi diagram as a preprocessing step takes $O(n \log n)$ time. Extracting the nominal configurations from the Voronoi diagram takes linear time. The number of vertices (nominal configurations) in a nominal configuration net is in the order of $O(n)$. Assuming that the maximum degree of a Voronoi diagram vertex is three, the edges in a nominal configuration net can be constructed in $O(n)$ time.

Conclusion Building the nominal configuration net in the global path planning layer takes $O(n \log n)$ time.

D. THE SPINE NET

1. Introduction

This is a graph consists of only straight line segments called spines. Two methods that generate this kind of roadmap are examined here. One is the well-known freeway method. The other is a novel sensor-oriented method proposed in this thesis. As a matter of fact, in the absence of a good algorithm to generate spine net automatically, some recent papers [Ref. 7] [Ref. 9] even based on hand-drawn spine nets as the starting point of investigations into path planning subject to κ_{max} constraint. It seems to suggest that the spine net has a natural appeal to layered path planning approach, and is more manageable when kinematic constraints are imposed on the solution path.

2. The Freeway Method

a. Background

The freeway method was first investigated by Brooks[Ref. 13] in 1983 as an empirical method to solve the basic motion planning problem. The intuition behind the method is similar to that of a Voronoi diagram, that is, keep the robot as far away as possible from the obstacles. But unlike Voronoi diagrams which require complex (but complete) method to construct and which consist of parabolic segments, the freeway method uses simple and intuitively appealing (but incomplete) method to extract only straight-line freeway spines from W.

This method is known to be incomplete, and hence may not always find a free path for the basic motion planning problem even if one exists. However, experiments with the freeway method have shown that it works fast in a relatively uncluttered workspace[Ref. 13].

b. Details of the Freeway Method

The spine net can be constructed using the freeway method as follows:[3]

Procedure Build Spine Net (Freeway Method):

1. **Step 1** Extract the Freeway Spines

   (a) For every pair of obstacle edges e_1 and e_2 facing each other, construct their bisector b.

   (b) Extend b outwards until it hits some other obstacles. See Figure 3.6(a).

   (c) Project the end points of e_1 and e_2 normally onto the bisector b. See Figure 3.6(b).

   (d) Discard those intervals of b which do not include at least a vertex from each of e_1 and e_2. See Figure 3.6(c).

   (e) Discard the portions of b which have clearance smaller than the radius of $\mathcal{D}$. That is, those portions which are not collision-free. See Figure 3.6(d).

   (f) The remaining intervals of b are the freeway spines between e_1 and e_2. There are at most two intervals per freeway spine.

2. **Step 2** Build the Spine Net $\mathcal{M}$

   (a) Compute the junctions (intersections) among the collection of freeway spines (pairwise) obtained from the last step.

   (b) Build the spine net $\mathcal{M} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the collection of all the directed turns ($< 180°$), and $\mathcal{E}$ the collection of the directed freeway spines.

The resulting spine net $\mathcal{M}$ is reminiscent of a Voronoi diagram. However, it consists of only rectilinear lines and lines of orientations $\pm 45°$. Note that

[3]This is a variant of Brook's method for the basic motion planning problem.

each spine has two directions, θ and $\theta + 180^{o}$. Every pair of intersecting spines generates four pairs of intersecting directed spines and eight distinct directed turns at each junction (see Figure 3.7).

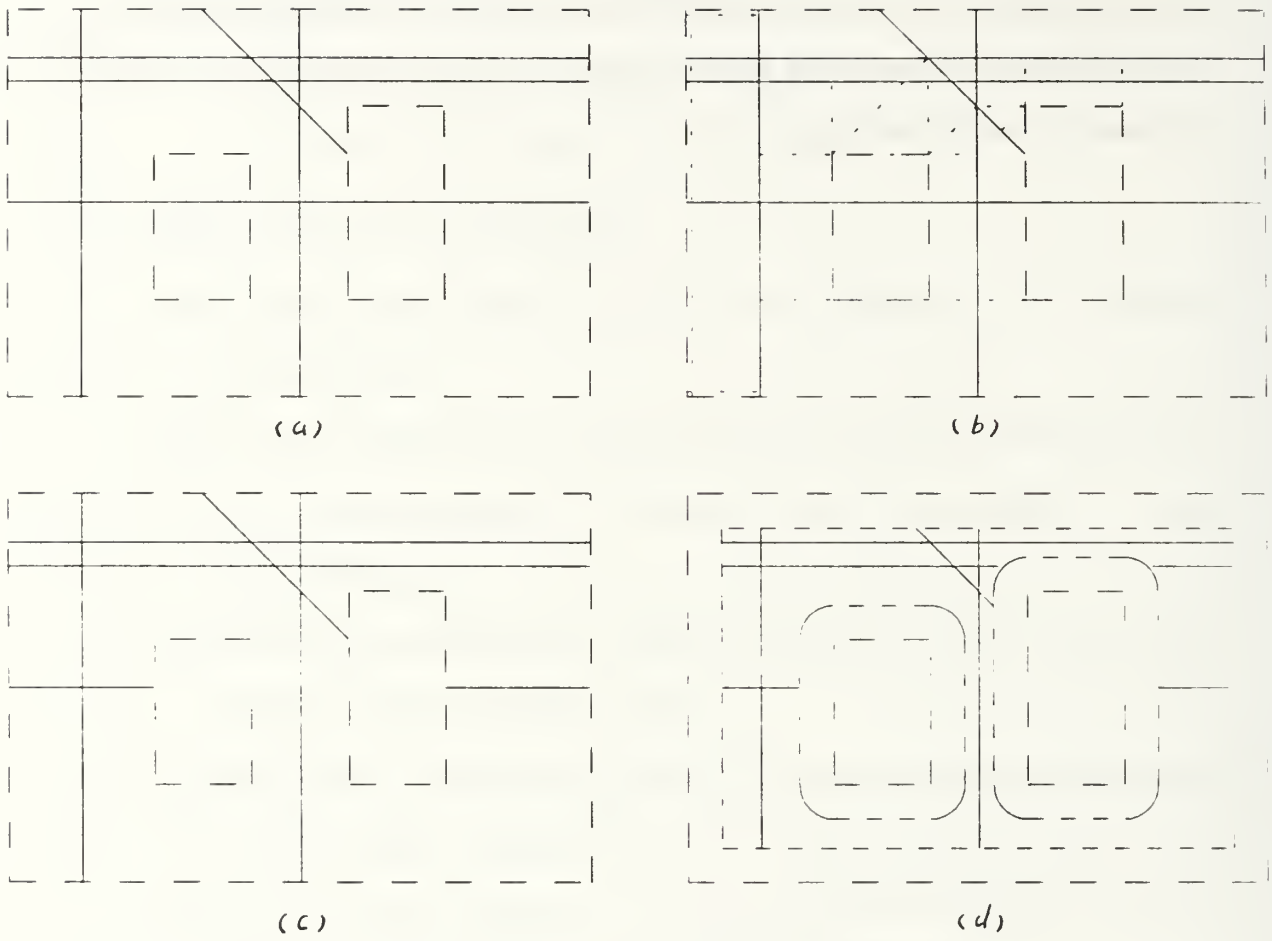


Figure 3.6: Construction of Freeway Spines

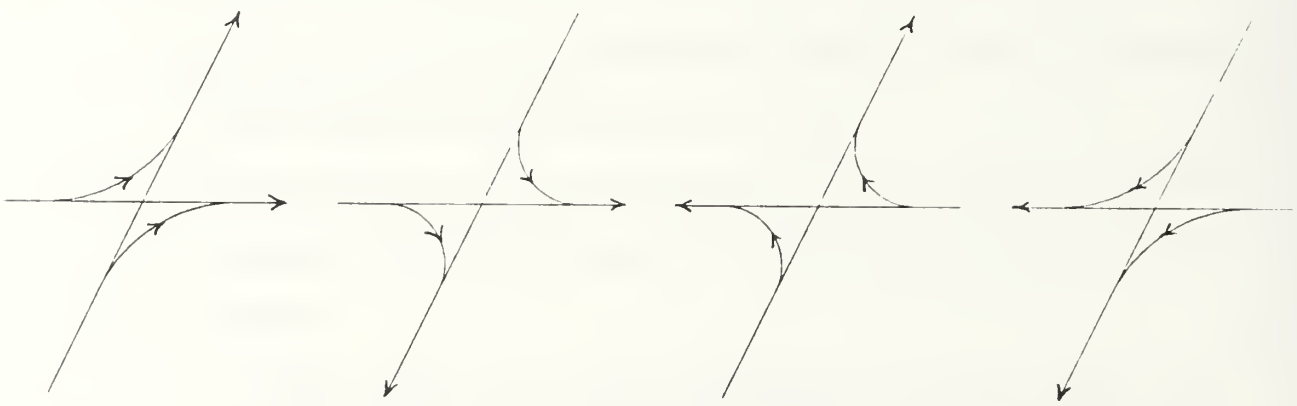


Figure 3.7: Intersecting Directed Spines and Directed Turns

3. The Sensor-Oriented Method

a. Background

The sensor-oriented method is a novel method proposed in this thesis to extract only rectilinear straight-line segments called spines from W. Intuitively, each spine is either situated at a suitable distance w_0 within within sensor (for example sonar) range and parallel to an obstacle wall, or is the bisector of two obstacle walls facing each other at a distance $< 2w_0$ and $> 2r_0$, where r_0 is the radius of the disc robot $\mathcal{D}$.

This method is based purely on the following observations and justifications:

- For a disc robot $\mathcal{D}$ that uses sonar sensors to correct its position and orientation errors, it is important that $\mathcal{D}$ follows the obstacle walls within the sonar range as far as possible.

  Supposing the sonar range is $[w_{min}, w_{max}]$, where $w_{min} > r_0$. w_0 is chosen to be the middle value of the sonar range for open walls, so that there are maximum error correction ranges at both sides of w_0. When two facing walls are closer than $2w_0$, it is still all right for $\mathcal{D}$ to follow the bisector line so long as it is wide enough for $\mathcal{D}$ to squeeze through. However, the narrower the passage, the more frequent the positional correction is required to carry out in order to avoid collision because of accumulated positional errors. When two facing walls are closer than $2w_{min}$, following the bisector may be possible but the sonar will be out of range.

- In order to avoid becoming a wall-following algorithm, this method extends the two ends of each spine outwards for connections with other spines. This allows for a choice between following walls at a greater distance, or taking shortcut

without a wall in range for position and orientation correction. The choice will be based on some cost functions to be discussed later in the graph search layer.

b. Details of the Sensor-Oriented Method

The spine net $\mathcal{M}$ can be constructed using the sensor-oriented method with the following procedure:

Procedure Build Spine Net (Sensor-Oriented Method):

1. **Step 1** Extract the Spines

   (a) For every obstacle wall e, construct a spine (line segment) parallel to it at a distance w_0.

   (b) Extend each spine outwards until it hits some other obstacles.

   (c) For every pair of obstacle walls e_1 and e_2 with the same orientation (either horizontal or vertical), if:

      i. their distance is $\leq 2r_0$, discard those portions of the spines of e_1 and e_2 within the common range.

      ii. their distance is $> 2r_0$ but $\leq r_0 + w_0$, replace those portions of the spines of e_1 and e_2 within the common range by the bisector of the two walls. Extend the bisector outwards until it hits some other obstacles or comes near to a parallel wall within the distance of r_0.

      iii. their distance is $> r_0 + w_0$ but $< w_{min} + w_0$, introduce an additional spine which is the bisector of the two walls. Extend the bisector outwards until it hits some other obstacles or comes near to a parallel wall within the distance of r_0. See Figure 3.8(a).

   (d) Truncate the two ends of each spine by a length of r_0, that is, those portions which are not collision-free. See Figure 3.6(b).

2. **Step 2** Build the Spine Net $\mathcal{M}$

   (a) Compute the junctions (intersections) among the collection of freeways (pairwise) obtained from the last step.

   (b) Build the roadmap of spine net $\mathcal{M} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ is the collection of all the directed turns ($= 90°$), and $\mathcal{E}$ the collection of all directed spines.

Note that each spine has two directions, θ and $\theta + 180°$, where $\theta = 0°$ or $90°$. Every pair of intersecting spines generates four pairs of intersecting directed spines and eight distinct directed turns at each junction (see Figure 3.7 for an illustration).

4. Data Structure of the Spine Net

Conceptually, a spine net consists of a collection of vertices which are directed turns, and a collection of edges which are sections of directed spines connecting two directed turns. In terms of data structure, a spine net $\mathcal{M} = (\mathcal{V}, \mathcal{E})$ can be constructed and represented as follows:

1. A spine is determined by the coordinates of its two end points. There are two directed spines associated with each spine.

2. The intersections between each pair of spines are calculated. If there are $O(m)$ spines, there will be $O(m^2)$ intersections. In the case of sensor-oriented spine net, $m = O(n)$, where n is the number of obstacle walls in W. In the case of freeway spine net, $m = O(n^2)$.

3. Each pair of intersecting spines generates eight distinct directed turns (vertices). Therefore, there are $O(m^2)$ vertices.

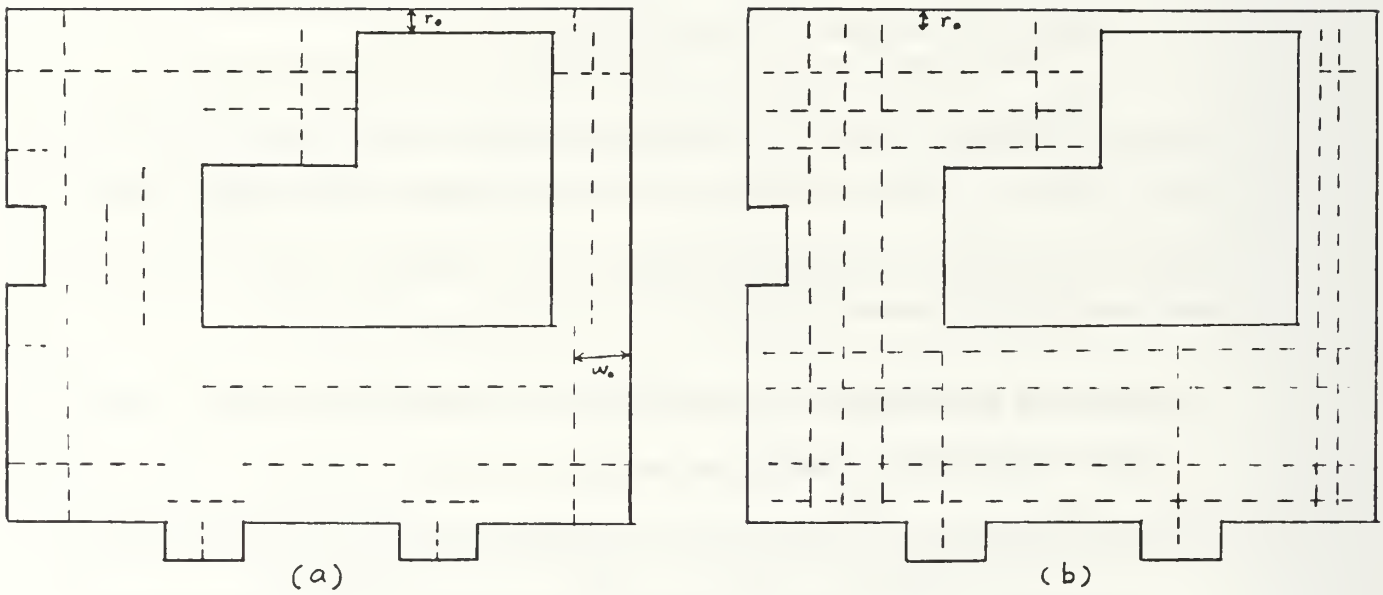


Figure 3.8: Construction of Spines Using Sensor-Oriented Method

4. Each vertex $\in \mathcal{V}$ can be represented as an exit point from a directed spine e_1 to another entry point on another intersecting directed spine e_2. In the case of the global spine net, the exit point is coincident with the entry point.

5. For every two vertices $v_1, v_2 \in \mathcal{V}$, if the entry point of v_1 lies ahead of the exit point of v_2 on the same directed spine, there is a directed edge $\in \mathcal{E}$ from v_1 to v_2. The directed edge is a segment of a directed spine, and is implicitly represented by the exit point and the entry point. There are potentially $O(m^4)$ edges.

5. Time Complexity Analysis of Building the Spine Net

1. Using the Freeway Method

   (a) *Extracting the Freeway Spines.* Assuming there are n obstacle edges in W. The number of potential bisectors is $O(n^2)$. Therefore, extracting the bisectors takes $O(n^2)$ time. Each bisector requires $O(n)$ time to check against intersection with obstacles and minimum clearance to become a freeway spine. Therefore, extracting the freeway spines takes a total of $O(n^3)$ time.

   (b) *Building the Spine Net.* Assuming that there are m freeway spines extracted. Theoretically, $m = O(n^2)$, although m is shown to be much less in number from empirical results. It takes $O(m^2)$ time to compute all directed turns (the vertices of $\mathcal{M}$) from the m freeway spines. Finding the edges takes $O(m^4)$ time. Therefore, building the spine net takes $O(m^4)$ time.

2. Using the Sensor-Oriented Method

(a) *Extracting the Spines.* Assuming there are n obstacle edges in W. The number of potential spines is $O(n)$. Therefore, extracting the spines takes $O(n)$ time. Each extended spines requires $O(n)$ time to check against intersection with obstacles. Therefore, extracting the spines takes a total of $O(n^2)$ time.

(b) *Building the Spine Net.* Assuming that there are m spines extracted, where $m = O(n)$. Using the same reasoning, building the spine net takes $O(m^4)$ time.

Conclusion Building the spine net in the global path planning layer takes $O(m^4)$ time, where $m = O(n^2)$ using the freeway method, or $m = O(n)$ using the sensor-oriented method.

IV. CUBIC SPIRALS

A. OVERVIEW

The cubic spiral is first investigated in [Ref. 12] by Y. Kanayama as a class of *smoothest* curves with respect to the cost function of $(d\kappa/ds)^2$. Only the section of a cubic spiral where curvature is null at both ends are used in motion planning. When such sections are connected to form a motion path, a path that is of class C^2 in its entirety can be obtained. Note that straight line segments are just special cases of cubic spirals. This makes cubic spirals a very useful tool in the local smooth path planning layer to smooth out all the turns in global roadmaps into class C^2.

In contrast, although straight line segments are also special cases of circular arcs, a path consisting of straight line segments and circular arcs is not C^2 in its entirety. This is because while straight line segments have null curvature at two ends, circular arcs have constant non-null curvatures. There is always a discontinuity in curvature when a straight line segment meets a circular arc.

There are other classes of curves that can be used in place of cubic spirals for planning smooth paths of class C^2. Examples are the clothoid curve [Ref. 14] and the B-spline curve [Ref. 15]. The Cubic spiral is chosen in this thesis because of its smoothness, elegance, and ease of use in local smooth path planning. This chapter introduces the properties of the cubic spiral before it is employed in the subsequent chapters of motion planning layers.

Some readers may find the Mathematical deductions in this chapter rather obscure, and may think otherwise about the elegance of cubic spirals. It must be pointed out that the basic properties of the unconstrained cubic spiral are not difficult. Additional Mathematical deductions are required in this chapter because of

the kinematic constraints imposed, and the inherent difficulty of smoothing out the turns of global roadmaps. Whatever curves one chooses for local smooth path planning purposes, the Mathematics required will always be much more than the basic properties of the chosen curve.

B. BASIC PROPERTIES OF CUBIC SPIRALS

While most people are familiar with the geometry of the circular arcs in the near-smooth path, they are not familiar with cubic spirals adopted in this thesis for smooth path planning. The readers are referred to [Ref. 12] for a good description of cubic spirals. This section only provides a quick summary of the important properties of cubic spirals related to path planning:

1. A cubic spiral with length l and maximum curvature $\kappa_{d,\alpha}$ (see Figure 4.1) is described by the following curvature function:

$$\kappa(s) = \kappa_{d,\alpha} \frac{s(l-s)}{l^2/4} \qquad 0 \leq s \leq l \tag{4.1}$$

2. Cubic spirals have no closed forms in Cartesian coordinates (see Figure 4.2). A point $(x, y) = (x(s), y(s))$ on a cubic spiral is given by,

$$x = x(s) = \int_0^s \cos\,\theta(t)\,dt$$
$$y = y(s) = \int_0^s \sin\,\theta(t)\,dt$$

Where the tangent direction function θ is given by,

$$x(s) = \int_0^s \kappa(t)\,dt \qquad 0 \leq s \leq l$$

Only the part of a cubic spiral between the two points s_1 and s_2 where curvature is null will be used in path planning.

3. Let $p_1 = (x_1, y_1, \theta_1)$ and $p_2 = (x_2, y_2, \theta_2)$ be two configurations with $(x_1, y_1) \neq (x_1, y_1)$. The pair of two configurations p_1 and p_2 are said to be *symmetric* (see Figure 4.3 for an illustration) if and only if,

$$\tan\left(\frac{\theta_1 + \theta_2}{2}\right) = \frac{y_2 - y_1}{x_2 - x_1} \tag{4.2}$$

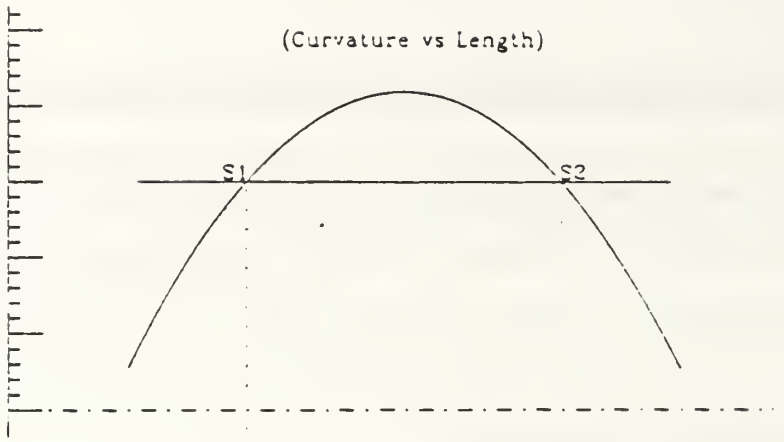


Figure 4.1: Curvature of Cubic Spiral

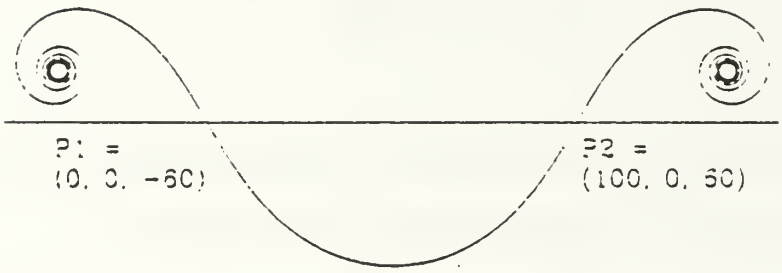


Figure 4.2: A Cubic Spiral Curve in Cartesian Coordinates

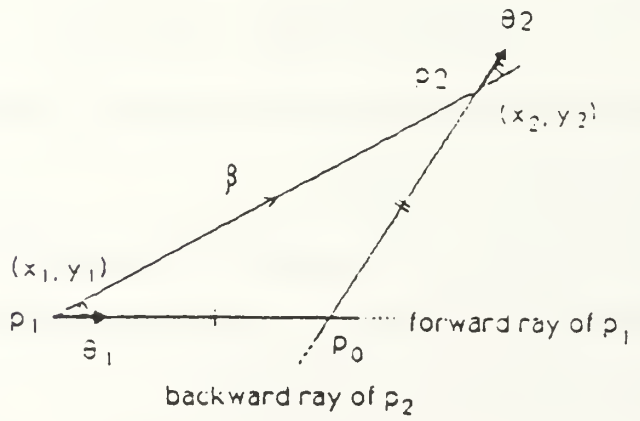


Figure 4.3: Symmetric Configurations

4. A cubic spiral is uniquely determined by a pair of symmetric configurations. In particular, it is uniquely determined by the *size* d and the *deflection* α of the symmetric pair which are defined as the Euclidean distance of the two points and the angle between the orientations (see Figure 4.4). In general, it is uniquely determined by any two of the following parameters:

   size d, deflection α, length l, height h, max curvature $\kappa_{d,\alpha}$

5. If two symmetric configurations are joined by two straight line segments (which are special cases of cubic spirals) and a cubic spiral (see Figure 4.5), then the whole path is of class C^2. In other words, there is curvature continuity throughout the whole path (except at the two path ends).

6. All cubic spirals with the same deflection α are similar to each other (see Figure 4.6). That is, for every two cubic spirals with the same α and with parameters $l_1, d_1, h_1, l_2, d_2, h_2$ respectively, the following relationship holds:

$$\frac{l_1}{l_2} = \frac{d_1}{d_2} = \frac{h_1}{h_2}$$

   If l_2 is chosen to be 1 unit, and the corresponding d_2 and h_2 at *deflection* α are denoted to be $D(\alpha)$ and $H(\alpha)$ respectively, then,

$$l = \frac{d}{D(\alpha)} = \frac{h}{H(\alpha)} \tag{4.3}$$

   Though without a closed form, $D(\alpha)$ can be obtained from the following integral equation,

$$D(\alpha) = 2\int_0^{1/2} \cos\left(\alpha\left[\frac{3}{2} - 2t^2\right]t\right)\, dt \tag{4.4}$$

   If a table is built for $D(\alpha)$ for $0° \leq \alpha \leq 360°$ (see Figure 4.7), the length l of a cubic spiral can be uniquely determined by d and α in discrete steps of the latter. Likewise applies to $H(\alpha)$.

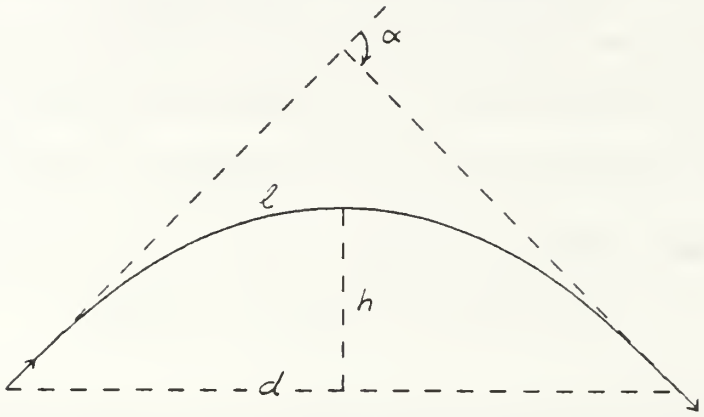


Figure 4.4: Parameters of Cubic Spirals

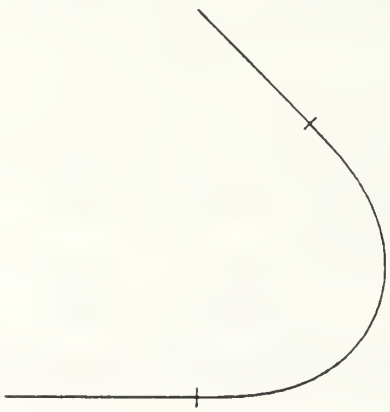


Figure 4.5: A C^2 Path Consisting of Two Line Segments and A Cubic Spiral

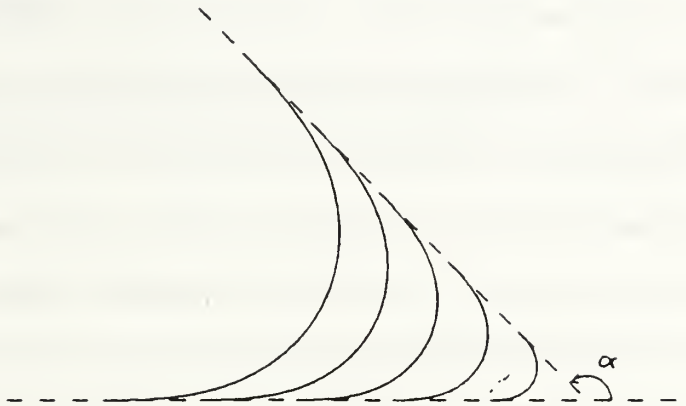


Figure 4.6: A Family of Similar Cubic Spirals with the same deflection α

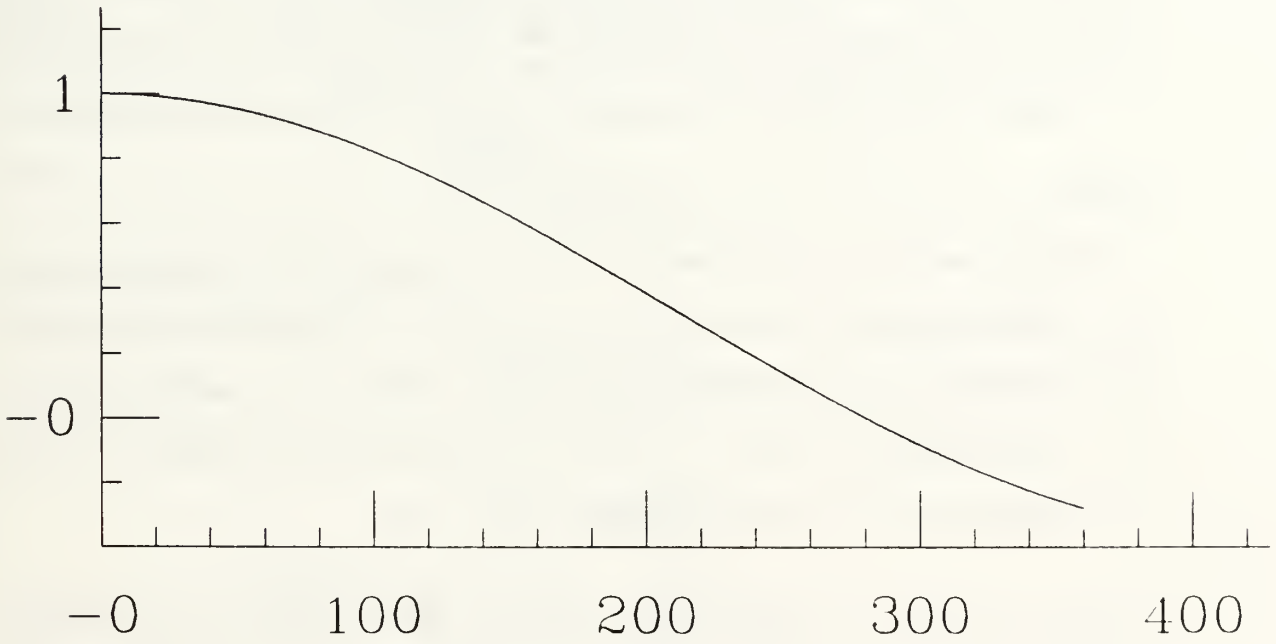


Figure 4.7: $D(\alpha)$ Plotted Against α in Steps of 1°

7. As seen from Figure 4.7, when α increases, $D(\alpha)$ decreases. $D(\alpha)$ equals to 0 when α is at about 281°. Since cubic spirals with $D(\alpha)$ less than or equal to 0 are difficult to handle, in this thesis, only those cubic spirals with deflection α less than or equal to 280° will be considered for path planning purposes.

8. The maximum curvature occurred at a cubic spiral determined by d and α is denoted as $\kappa_{d,\alpha}$ (note: this is different from the kinematic constraint κ_{max} imposed by the robot). Because of the symmetry of the cubic spiral, it occurs at the mid point of the curve where $s = l/2$ (see Figure 4.1). From Equation 4.1, we have

$$\alpha = \int_0^l \kappa(s)\, ds = \frac{2\kappa_{d,\alpha} l}{3}$$

Together with Equation 4.3, we have

$$\kappa_{d,\alpha} = \frac{3\alpha}{2l} = \frac{3\alpha D(\alpha)}{2d} \tag{4.5}$$

or

$$d = \frac{3\alpha D(\alpha)}{2\kappa_{d,\alpha}} \tag{4.6}$$

9. The maximum rate of change of curvature occurred at a cubic spiral determined by d and α is denoted as $\dot{\kappa}_{d,\alpha}$ (note: this is different from the kinematic constraint $\dot{\kappa}_{max}$ imposed by the robot). It occurs at the beginning and the end of the curve where $s = 0$ and $s = l$ respectively (see Figure 4.1). From Equation 4.1 , 4.3 and 4.5, we have

$$\dot{\kappa}_{d,\alpha} = \left|\frac{d\kappa}{ds}\right|_{s=0} = \frac{4\kappa_{d,\alpha}}{l} = \frac{6\alpha}{l^2} = \frac{6\alpha D^2(\alpha)}{d^2} \tag{4.7}$$

C. CIRCUMSCRIBING CUBIC SPIRALS AND INSCRIBING CIRCLES

This section studies further properties of cubic spirals.

Notation An angle α in this chapter is assigned the unit degree (°) or radian depending on the context it is in. The unit *degree* is assumed when α is used in the context of plotting curves in steps of 1°, or is subject to a range restriction. This is because there are angles of special interests, for example 280°, which can not be conveniently expressed in radians. On the other hand, the unit *radian* is assumed when α appears in Mathematical formulas. The readers should have no problem in making the distinction for a given context.

Proposition 4.1 *Given a cubic spiral uniquely determined by a pair of symmetric configurations p_1 and p_2 with size d (> 0) and deflection α ($\leq 280°$), we can uniquely determine a circle with radius $r_{d,\alpha}$ that inscribes the cubic spiral at the two configurations (see Figure 4.8). Inversely, we say that the cubic spiral circumscribes the circle at these two configurations. We call them the "inscribing circle" and the "circumscribing cubic spiral" respectively. From geometry of the circle, we have,*

$$r_{d,\alpha} = \frac{d}{2\sin(\alpha/2)} \tag{4.8}$$

Together with Equation 4.6, we have,

$$r_{d,\alpha} = \frac{3D(\alpha)}{2\kappa_{d,\alpha}} \left(\frac{\alpha/2}{\sin(\alpha/2)} \right) \tag{4.9}$$

or

$$\kappa_{d,\alpha} = \frac{3D(\alpha)}{2r} \left(\frac{\alpha/2}{\sin(\alpha/2)} \right) \tag{4.10}$$

Proposition 4.2 *Any two points on a circle with orientations tangent to the circle and equal in direction are a symmetric pair. Therefore, they can be circumscribed by a cubic spiral (see Figure 4.8). Let r be the radius of the circle, $d(> 0)$ and $\alpha(\leq 280°)$ be the size and deflection of the symmetric configurations. The maximum curvature incurred at the circumscribing cubic spiral, denoted by $\kappa_{d,\alpha}$ can be determined by any two of the above three parameters (r, d and α). See Equation 4.5 for that determined by d and α, and Equation 4.10 for that determined by r and α.*

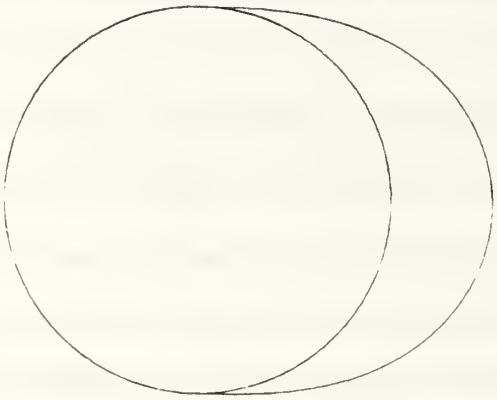


Figure 4.8: A Circle Inscribing a Cubic Spiral

Proposition 4.3 *Denote by l the length of the cubic spiral with size d and deflection α circumscribing the circle of radius r. From Equation 4.5 and 4.9, we have,*

$$l = \frac{3\alpha}{2\kappa_{d,\alpha}} = \frac{6\alpha r \sin(\alpha/2)}{6D(\alpha)(\alpha/2)} = 2r\left(\frac{\sin(\alpha/2)}{D(\alpha)}\right)$$

Let

$$F_1(\alpha) = \frac{\sin(\alpha/2)}{D(\alpha)}$$

Figure 4.9 shows $F_1(\alpha)$ plotted against α in discrete steps of $1°$. It shows that,

$$\text{as } \alpha \to 281°, F_1(\alpha) \to \infty, \text{ therefore } l \to \infty$$

This further justifies the restriction to consider only cubic spirals of deflection $\alpha \leq 280°$. In fact, for all practical purposes, only cubic spirals of $\alpha \ll 280°$ need to be considered. Proposition IV.11 in section E. shows a method of avoiding cubic spirals with large deflections.

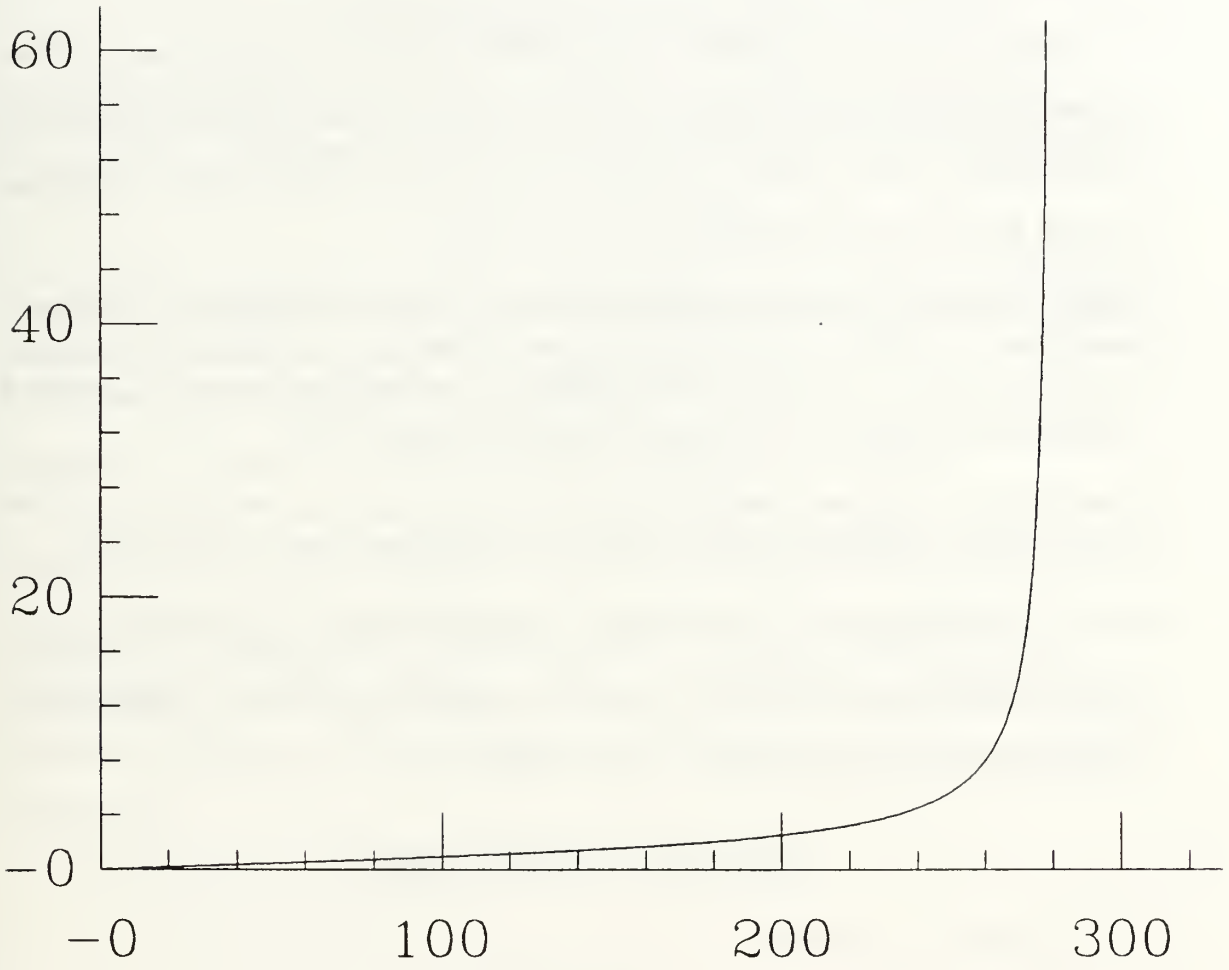


Figure 4.9: $F_1(\alpha)$ Plotted Against α in Steps of 1°

D. CUBIC SPIRALS SUBJECT TO κ_{max} AND $\dot{\kappa}_{max}$ CONSTRAINTS

The previous sections illustrate the general properties of the cubic spiral, and the maximum $\kappa_{d,\alpha}$ and $\dot{\kappa}_{d,\alpha}$ incurred at each cubic spiral. In this section, the properties of the subset of cubic spirals which meets the gloabl constraints of κ_{max} and $\dot{\kappa}_{max}$ imposed by the disc robot will be studied in detail.

Proposition 4.4 *Given two intersecting lines e_1 and e_2 with deflection α measured from e_1 to e_2, and $0° \leq \alpha \leq 280°$. Ignoring κ_{max} and $\dot{\kappa}_{max}$ constraints, we can obtain a family of similar cubic spirals with deflection α connecting any pair of symmetric configurations lying on e_1 and e_2 respectively (see Figure 4.6).*

Question 4.1 *If the cubic spiral is subject to κ_{max} and $\dot{\kappa}_{max}$ constraints, find the subset of this α-deflection family of cubic spirals that can meet the constraints.*

For a cubic spiral of size d in this α-deflection family, denote its maximum curvature and maximum rate of change of curvature by $\kappa_{d,\alpha}$ and $\dot{\kappa}_{d,\alpha}$ respectively. If it meets the curvature constraints, we must have,

$$\kappa_{d,\alpha} \leq \kappa_{max} \quad and \quad \dot{\kappa}_{d,\alpha} \leq \dot{\kappa}_{max}$$

From Equation 4.5, we have

$$\kappa_{d,\alpha} = \frac{3\alpha D(\alpha)}{2d} \leq \kappa_{max}$$

or

$$d \geq \frac{3\alpha D(\alpha)}{2\kappa_{max}} \tag{4.11}$$

From Equation 4.7, we have

$$\dot{\kappa}_{d,\alpha} = \frac{6\alpha D^2(\alpha)}{d^2} \leq \dot{\kappa}_{max}$$

or

$$d \geq D(\alpha)\sqrt{\frac{6\alpha}{\dot{\kappa}_{max}}} \tag{4.12}$$

From Equation 4.11 and 4.12, we have,

$$d \geq \max\left(\frac{3\alpha D(\alpha)}{2\kappa_{max}}, D(\alpha)\sqrt{\frac{6\alpha}{\dot{\kappa}_{max}}}\right) \tag{4.13}$$

Proposition 4.5 *The subset of the α-deflection cubic spiral family that meets the κ_{max} and $\dot{\kappa}_{max}$ constraints are those with size d such that d meets the condition in Equation 4.13.*

Definition 4.1 *If a circle meets the condition that "every two symmetric configurations on its circumference with deflection α ($\leq 280°$) can be circumscribed by a cubic spiral that meets the κ_{max} constraint", we say that the circle is fully κ_{max}-admissible. Likewise for a circle that is fully $\dot{\kappa}_{max}$-admissible.*

Question 4.2 *Find the set of circles that are fully κ_{max}-admissible.*

Suppose we have a circle of radius r. Every two symmetric configurations on the circumference of the cirle with deflection α can always be circumscribed by a cubic spiral without curvature constraints. Denote the maximum curvature and maximum rate of change of curvature of the circumscribing cubic spiral by $\kappa_{d,\alpha}$ and $\dot{\kappa}_{d,\alpha}$ respectively.

If the circle is κ_{max}-admissible, then we must have,

$$\kappa_{d,\alpha} \leq \kappa_{max}$$

Combining with Equation 4.9, we have,

$$r \geq \frac{3D(\alpha)}{2\kappa_{max}}\left(\frac{\alpha/2}{\sin(\alpha/2)}\right) \tag{4.14}$$

Let

$$r_{min} = \max_{0° < \alpha \leq 280°}\left(\frac{3D(\alpha)}{2\kappa_{max}}\left[\frac{\alpha/2}{\sin(\alpha/2)}\right]\right) = \frac{3}{2\kappa_{max}}\max_{0° < \alpha \leq 280°}\left(\frac{D(\alpha)(\alpha/2)}{\sin(\alpha/2)}\right)$$

and let

$$F_2(\alpha) = \frac{D(\alpha)(\alpha/2)}{\sin(\alpha/2)}$$

We can easily see that the upper bound of $F_2(\alpha)$ is 1,

$$\max_{0°<\alpha\leq 280°} F_2(\alpha) = \left(\max_{0°<\alpha\leq 280°} D(\alpha)\right)\left(\max_{0°<\alpha\leq 280°} \frac{\alpha/2}{\sin(\alpha/2)}\right) = 1 \times 1 = 1$$

Figure 4.10 shows $F_2(\alpha)$ plotted against α in discrete steps of 1°. It confirms that,

$$\max_{0°<\alpha\leq 280°} F_2(\alpha) = \lim_{\alpha\to 0°} F_2(\alpha) = 1$$

Therefore,

$$r_{min} = \frac{3}{2\kappa_{max}} \tag{4.15}$$

Proposition 4.6 *All circles with radius $r \geq r_{min}$ are fully κ_{max}-admissible.*

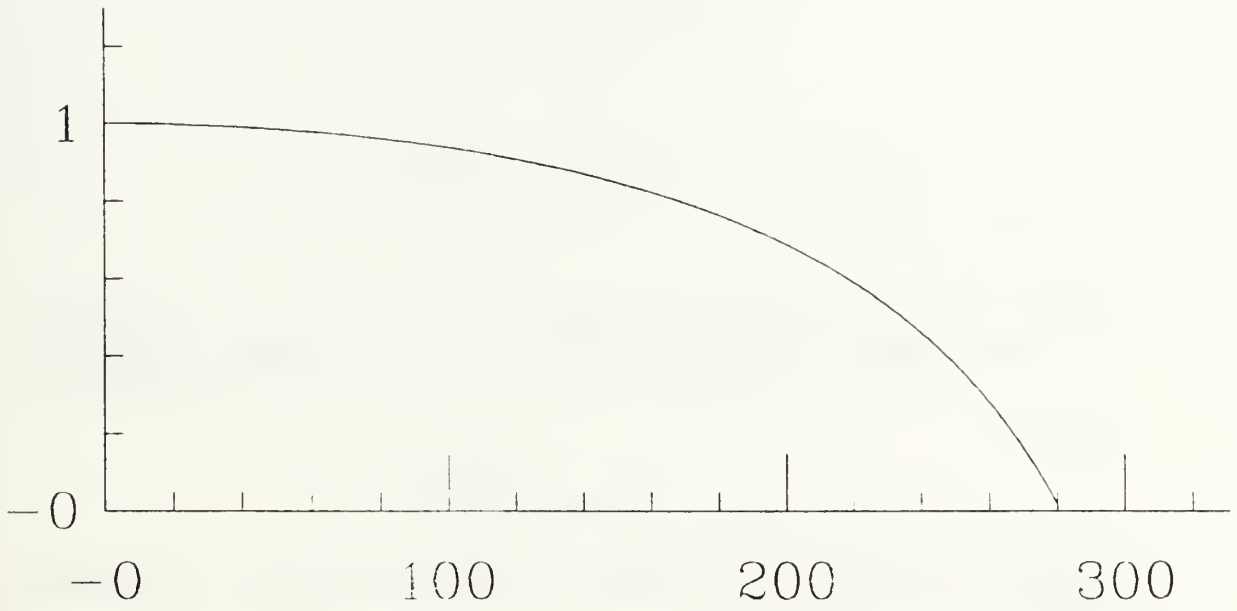


Figure 4.10: $F_2(\alpha)$ Plotted Against α in Steps of 1°

Question 4.3 *Find the set of circles that are fully $\dot{\kappa}_{max}$–admissible.*

If the circle is $\dot{\kappa}_{max}$–admissible, then we must have,

$$\dot{\kappa}_{d,\alpha} \leq \dot{\kappa}_{max}$$

From Equations 4.7 and 4.6, we have,

$$\dot{\kappa}_{d,\alpha} = \frac{6\alpha D^2(\alpha)}{4r^2 \sin^2(\alpha/2)} \leq \dot{\kappa}_{max}$$

or

$$r \geq \sqrt{\frac{3}{2\dot{\kappa}_{max}}} \left[\frac{\sqrt{\alpha} D(\alpha)}{\sin(\alpha/2)}\right] \tag{4.16}$$

Let

$$r'_{min} = \max_{0° < \alpha \leq 280°} \left(\sqrt{\frac{3}{2\dot{\kappa}_{max}}} \left[\frac{\sqrt{\alpha} D(\alpha)}{\sin(\alpha/2)}\right]\right) = \sqrt{\frac{3}{2\dot{\kappa}_{max}}} \left(\max_{0° < \alpha \leq 280°} \left[\frac{\sqrt{\alpha} D(\alpha)}{\sin(\alpha/2)}\right]\right)$$

and let

$$F_3(\alpha) = \frac{\sqrt{\alpha} D(\alpha)}{\sin(\alpha/2)}$$

Figure 4.11 shows $F_3(\alpha)$ plotted against α in discrete steps of 1°. It shows that,

$$\text{as } \alpha \to 0, \quad F_3(\alpha) \to \infty, \quad \text{therefore } r'_{min} \text{ is unbounded}$$

Proposition 4.7 *No circles is fully $\dot{\kappa}_{max}$–admissible.*

Definition 4.2 *Given an angle β where $0° < \beta \leq 280°$. If a circle meets the condition that "every two symmetric configurations on its circumference with deflection α, where $\beta \leq \alpha \leq 280°$, can be circumscribed by a cubic spiral that meets the $\dot{\kappa}_{max}$ constraint, but not those with deflection $\alpha < \beta$", we say that the circle is partially $\dot{\kappa}_{max}$–admissible at β.*

Question 4.4 *For the circle with radius r_{min} that is fully κ_{max}–admissible, find β which makes it partially $\dot{\kappa}_{max}$–admissible at β.*

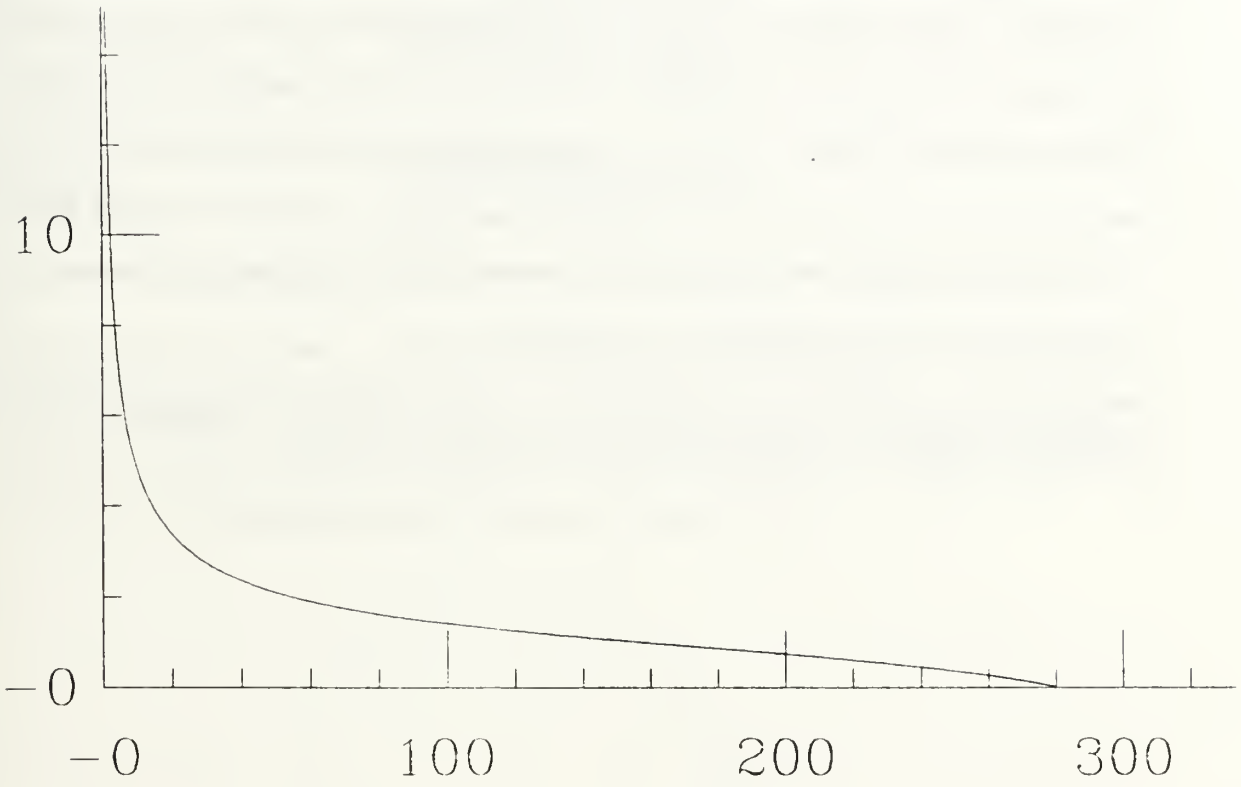


Figure 4.11: $F_3(\alpha)$ Plotted Against α in Steps of 1°

From Equation 4.16, we can see that β occurs at the point where,

$$r_{min} = \sqrt{\frac{3}{2\dot{\kappa}_{max}}} \left(\frac{\sqrt{\beta} D(\beta)}{\sin(\beta/2)} \right) = \sqrt{\frac{3}{2\dot{\kappa}_{max}}} F_3(\beta)$$

Together with Equation 4.15, we have

$$\frac{3}{2\kappa_{max}} = \sqrt{\frac{3}{2\dot{\kappa}_{max}}} F_3(\beta)$$

or

$$\beta = F_3^{-1} \left(\frac{\sqrt{(3/2)\dot{\kappa}_{max}}}{\kappa_{max}} \right) \tag{4.17}$$

There is no closed form solution for β; we can only find an approximate value for β from the curve of $F_3(\alpha)$ as shown in Figure 4.11.

Proposition 4.8 *The circle with radius r_{min} that is fully κ_{max}–admissible is partially $\dot{\kappa}_{max}$–admissible at β as determined in Equation 4.17.*

E. BOUNDING REGIONS OF CUBIC SPIRALS

Since there is no closed form for cubic spirals, it is not possible to check for collisions between cubic spirals and the polygonal obstacles in W using algebraic equations. This section attempts to approximate a cubic spiral by a bounding region such that the cubic spiral lies completely within the bounding region. The edges of the bounding region will consist of only straight line segments and circular arcs. Instead of checking the cubic spiral against the obstacles, we will check the bounding region instead which allows algebraic solutions. The smaller the bounding region, the closer the approximation and hence the more optimal the resulting solution.

Proposition 4.9 *Given a cubic spiral uniquely determined by a pair of symmetric configurations p_1 and p_2 with size d (> 0) and deflection α ($< 180°$), we can construct a bounding region BR_1 for it from the two lines generated by p_1 and p_2 and the circular arc that inscribes the cubic spiral at the two points (see Figure 4.12). The radius $r_{d,\alpha}$ of the inscribing circle is given by Equation 4.8.*

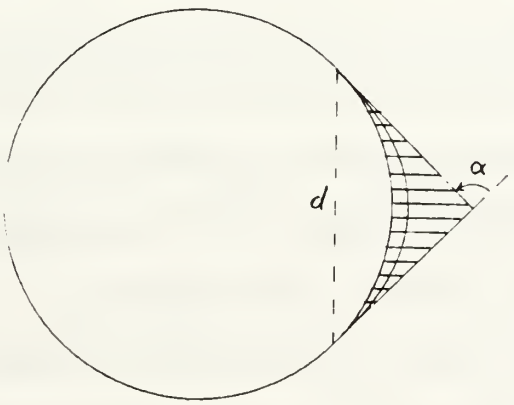


Figure 4.12: Bounding Region BR_1 of a Cubic Spiral

As α increases, so does the area of BR_1 of the cubic spiral. When $\alpha \geq 180°$, BR_1 is undefined. We need an alternative way of defining the bounding region when α is large.

Proposition 4.10 *Given a cubic spiral uniquely determined by a pair of symmetric configurations p_1 and p_2 with size d (> 0) and deflection α ($\leq 280°$), we can construct a bounding region BR_2 for it from the two lines generated by p_1 and p_2, its inscribing circular arc, and the tangential line at the middle of the curve at a height h from the base (see Figure 4.13). The height h can be obtained from Equation 4.3 as,*

$$h = \frac{dH(\alpha)}{D(\alpha)} = \frac{3\alpha\, H(\alpha)}{2\kappa_{d,\alpha}} \tag{4.18}$$

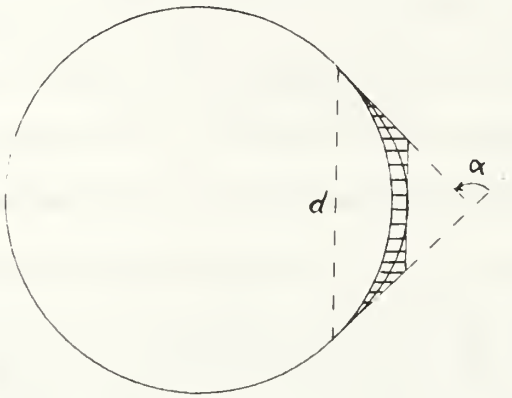


Figure 4.13: Bounding Region BR_2 of a Cubic Spiral

As seen from Figure 4.7, when α goes beyond 281°, $D(\alpha)$ becomes negative. The two legs of the cubic spiral cross and BR_2 is ill-defined. This justifies the restriction to consider cubic spirals of deflection $\alpha \leq 280°$. In the next chapter, it will be shown that this restriction is sufficient for local smooth path planning.[1] Hence the bounding region BR_2 is sufficient for use in most cases.

As α increases, so does the area of BR_2 of the cubic spiral. Although the area of BR_2 is always smaller than that of BR_1, and is thus a better approximation for the cubic spiral, it can still be excessive for large α. The following proposition develops a technique to get around this problem. It also gets around the problem of making a turn of deflection $\alpha > 280°$ using cubic spirals.

[1]This is not true for the end portions path planning, which requires breaking a cubic spiral into a sequence of sub cubic spirals.

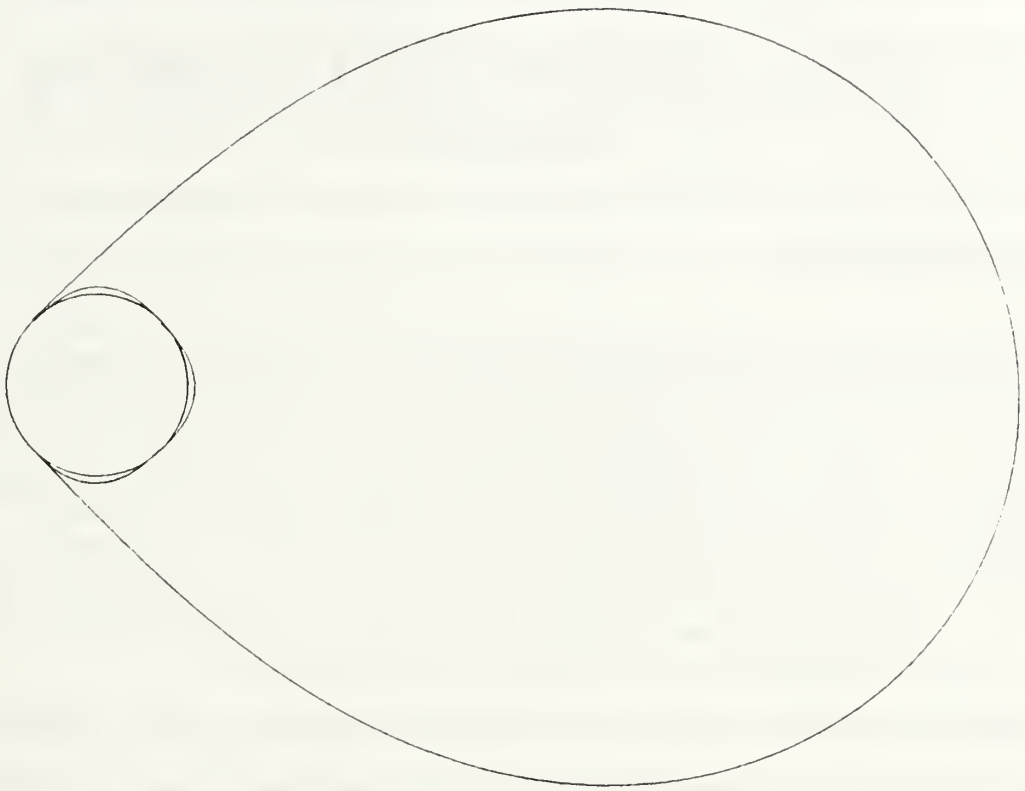


Figure 4.14: Substituting a Cubic Spiral with a Sequence of Sub-Spirals

Proposition 4.11 *Given two configurations p_1 and p_2 on a κ_{max}-admissible circle with size d (> 0) and deflection α ($\leq 360°$), we can always join them by a sequence of n cubic spirals, each of deflection $\alpha_i = \alpha/n$ and satisfying the κ_{max} constraint, and the sequence is of class C^2 in its entirety. The bounding region of this sequence of n cubic spirals is the union of the bounding regions of each sub cubic spiral, which is smaller in area than the bounding region of the original cubic spiral (see Figure 4.14).*

Proof Simply divide the circular arc between p_1 and p_2 into $n-1$ equal parts separated by $q_1 \cdots q_{n-2}$. Connect $p_1q_1, q_1q_2, \cdots, q_{n-2}p_2$ by cubic spirals. It can be easily shown that each cubic spiral is of deflection $\alpha_i = \alpha/n$. Since the inscribing circle is κ_{max}-admissible, all the n sub cubic spirals meet the κ_{max} constraint. In general, α_i need not be of equal size. They need only be small enough, say $\leq 90°$, to produce a convenient sequence of sub-spirals.

V. LOCAL SMOOTH PATH PLANNING LAYER

A. OVERVIEW

The local smooth path planning layer is the second layer in the layered motion planning approach proposed in this thesis. The goal of this layer is to modify the coarse global roadmap obtained from the first layer into a *smooth roadmap*. A smooth roadmap is one where every path in it is smooth. This layer is also a preprocessing step that needs to be carried out only once together with the global path planning layer to serve all subsequent on-line queries of motion planning.

Various kinds of curves could be used for smoothing purposes. This thesis chooses a class of curves called cubic spirals. Since straight line segments are just special cases of cubic spirals, a smooth roadmap is said to consist of only cubic spirals.

Regardless of which kind of smooth curve is chosen, this layer (and the end portions path planning layer to be discussed in the next chapter) faces the same challenges as follows:

1. The smooth curve that is adopted for a turn, or is replacing an original circular turn, must be collision-free. There are two sub-tasks involved here: one is to find plausible candidates in polynomial time, another is to check them against obstacles. So far, none of the smooth curves that have nice properties in curvatures has closed form in Cartesian coordinates. If we fixed the number of incremental steps, tracing out an approximate trajectory of the smooth curve can be achieved in polynomial time. However, this only helps in collision

checking, it does not facilitate candidates generation. This thesis proposes the bounding region concept described in Chapter IV to tackle this problem.

2. The smooth curve is subject to two kinematic constraints: κ_{max} and $\dot{\kappa}_{max}$. This severely reduces the number of candidates and the chances of completing a turn with smooth curves. This problem is most evident in the case of visibility net where the concern must be brought up during the global path planning layer to increase the chances of success. Modifications required for the construction of the visibility net will be discussed in detail in this chapter. In addition, additional Mathematics are required to study the properties of the subset of the smooth curves which meet the kinematic constraints. Such Mathematics are normally tedious and non-trivial as can be seen in Chapter IV.

Although only cubic spirals are used in this thesis, different techniques are required for smoothing different classes of roadmaps. In this thesis, roadmaps are classified according to their geometric properties, not the methods from which they are generated. Roadmaps in the same class will have similar basic data structure representation. At present, three kinds of roadmaps have been classified in the global path planning layer, namely, the visibility net, the nominal configuration net and the spine net.

In the following sections, the smoothing techniques for each class of roadmaps will be investigated and their time complexities analyzed. In some cases, modifications to the data structure are required for the smooth roadmap.

B. SMOOTHING TECHNIQUE FOR THE VISIBILITY NET

1. Shortcomings of the Unconstrained Visibility Net

At a glance, smoothing a visibility net seems trivial: just replace each circular turn by a cubic spiral turn, and then check for collision. This is true if there is no concern of the success rate of this operation. If the visibility net is built in the global path planning layer without considering the kinematic constraints, many of the circular turns may not be replaceable by constrained cubic spirals. In fact, if it is built with $r_0 \ll r_{min}$, none of the circular turns may be smoothed out.

Such complications of the visibility net are due to the peculiarity of the visibility graph method, where the turns are planned as close to the corners as possible. As a result, the constraints of the smooth path must be considered at the global path planning layer. Otherwise, the unconstrained visibility net may be too "tight" to allow for local smooth path planning. This problem is not faced by the spine net, where the turns are planned at moderate distance (not as far as possible) from a turning corner. There are "normally" enough allowance between the turns and the corners for local smooth path planning in the spine net.

In the following discussions, the visibility net will be rebuilt subject to the κ_{max} constraint first, and then the $\dot{\kappa}_{max}$ constraint next. Thereafter, a not-so-straight-forward smoothing technique will be proposed to increase the success rate of this layer.

2. Rebuilding the Visibility Graph Subject to κ_{max} Constraint

In the unconstrained visibility graph $\mathcal{M}$, the visible-directed-circular-arcs (from now on abbreviated as circular arcs) are of radius r_0, which is the radius of $\mathcal{D}$.

From proposition IV.6 and equation 4.15, if

$$r_0 \geq r_{min} = \frac{3}{2\kappa_{max}}$$

then all circular arcs can be circumscribed by cubic spirals satisfying the κ_{max} constraint. However, if $r_0 \ll r_{min}$, then most, if not all, of the circular arcs will not be replaceable by constrained cubic spirals.

A first-cut solution seems easy; just grow the corners of the obstacles in W by r_{min} instead of r_0. Unfortunately, there are three problems with this solution:

1. Where $\mathcal{D}$ can pass by a corner with clearance r_0, the new bumper of $r_{min} > r_0$ forces $\mathcal{D}$ to make an unnecessary turn (see Figure 5.1(a)).

2. When $\mathcal{D}$ makes a turn about a corner, it needs only maintain a clearance of r_0 at its closest point to the corner (see Figure 5.1(b)). Growing the corner by r_{min} is too excessive.

3. By growing the corners by $r_{min} > r_0$, some of the channels may become inaccessible (see Figure 5.1(c)), while in fact $\mathcal{D}$ can pass through them without difficulty (see Figure 5.1(d)).

A better solution is to construct $\mathcal{M} = (\mathcal{V}, \mathcal{E})$ with the following steps:

Procedure Build Visibility Net 2:

1. Denote the original convolution diagram by $\mathcal{C}$. Construct another convolution diagram $\mathcal{C}'$ by circumscribing each circular vertex in $\mathcal{C}$ at its tip with a circle of radius r_{min} (see Figure 5.2).

2. Construct $\mathcal{V}$ and $\mathcal{E}$ based on $\mathcal{C}'$ with the exception that visibility of an edge in $\mathcal{E}$ is checked against $\mathcal{C}$, not $\mathcal{C}'$. This will ensure that a disc robot $\mathcal{D}$ need not make an unnecessary turn when it can pass by a corner in a straight line.

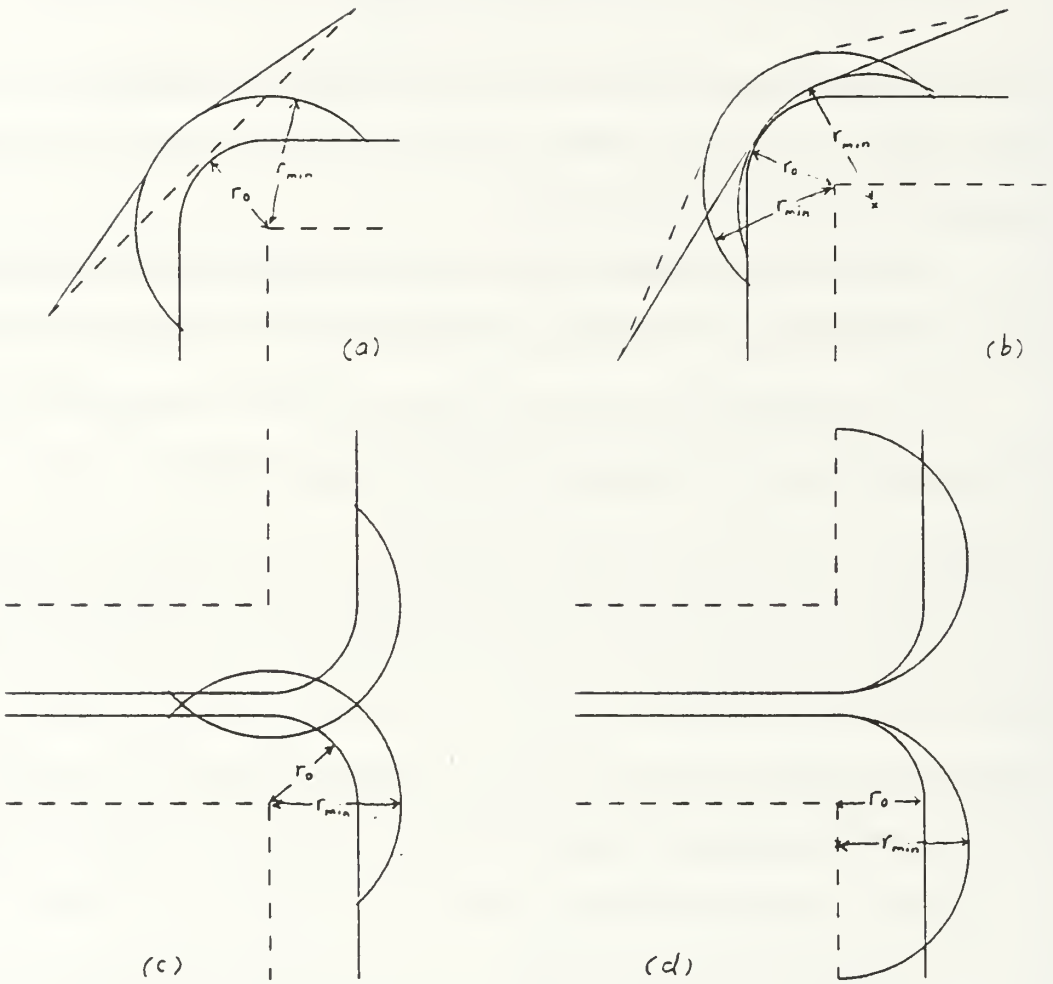


Figure 5.1: Problems of Growing Obstacle Corners by $r_{min} > r_0$

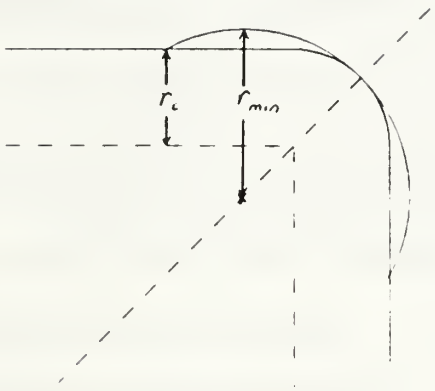


Figure 5.2: Circumscribing Circular Vertex with Circle of radius r_{min}

Where there are narrow passages, $\mathcal{C}'$ may still show that it is not possible to construct a circular vertex (see Figure 5.1(c)) while in fact it is possible. The trick is to circumscribe the original circular vertex at a point that can circumvent the constraints of the environment (see Figure 5.1(d)). However, such a fine point of optimality will not be dealt with in this thesis.

3. Rebuilding the Visibility Graph Subject to $\dot{\kappa}_{max}$ Constraint

The $\dot{\kappa}_{max}$ constraint is the most irritating problem in constructing the visibility graph. The reason is simply because there is no circles that is fully $\dot{\kappa}_{max}$-admissible (see proposition IV.7). Even if $r_0 \gg r_{min}$, this problem still persists in non-trivial cases. In the following discussions, *Procedure Build Visibility Net 2* will be modified to cater for the additional $\dot{\kappa}_{max}$ constraint.

Let $r_{min} = \max(r_{min}, r_0)$. According to proposition IV.8, there is a β where all turns of angle $\geq \beta$ can also satisfy the $\dot{\kappa}_{max}$ constraint. Denote the size of the cubic spiral at deflection β as d_β, and the chord supporting it $chord_\beta$. The remaining problem is to handle turns with deflection $\alpha < \beta$ subject to the $\dot{\kappa}_{max}$ constraint.

From equation 4.12, we know that all cubic spirals with size

$$d_\alpha \geq D(\alpha)\sqrt{\frac{6\alpha}{\dot{\kappa}_{max}}}$$

can meet the $\dot{\kappa}_{max}$ constraint. Since $F_4 = \sqrt{\alpha}D(\alpha)$ is a non-decreasing function in the range $0° \leq \alpha \leq 109°$ (see Figure 5.3), it follows that

$$d_\alpha < d_\beta \ , \quad \text{where } 0° \leq \alpha < \beta \leq 109°$$

That is, for every turn of angle $\alpha < \beta$, if we find another inscribing circle with chord length d_β, we can find a corresponding cubic spiral that can meet the $\dot{\kappa}_{max}$ constraint (see Figure 5.4(b)). Also, since the radius of this circle is clearly greater than r_{min}, the cubic spiral can meet the κ_{max} constraint as well.

Unfortunately, the problem has not been fully resolved yet at this point, which makes this approach very sticky. First, the arc of the new circle which is the bounding edge of the cubic spiral, intersects the circle with radius r_{min}. This means that the bounding region will collide with the corners of the C–obstacles. One solution is to grow the r_{min} circle by a bumper of Δr, where the $chord_\beta$ of the circle of radius $r_\beta = r_{min} + \Delta r$ just touches the r_{min} circle (see Figure 5.4(a)). Note that the r_β circle is concentric with the r_{min} circle, not tangential. From geometry, we have

$$r_\beta = \frac{r_{min}}{\cos(\beta/2)} \tag{5.1}$$

or

$$\Delta r = r_\beta - r_{min} = r_{min}(1 - \sec(\beta/2)) \tag{5.2}$$

Second, if two turns of angle $< \beta$ are very close, the β–chords of both may intersect each other, thus no solution for collision-free cubic spirals for them (see Figure 5.5).

There is clearly a solution for them, and it will be preposterous of this procedure to declare no solution at this point. One not-very-elegant way to get around this problem is to create a third r_β circle further down the road to make room for two β–chords (see Figure 5.6) just for these two circular vertices. All vertices lying on the second r_β circle that are too close to the first r_β circle are removed and recomputed on the third r_β circle.

With these additional considerations for $\dot{\kappa}_{max}$ constraint, the visibility net $\mathcal{M}$ can be rebuilt using *Procedure Build Visibility Net 2* with all the necessary modifications.

4. The Smoothing Technique

After rebuilding the visibility net taking into accounts the kinematic constraints, what is left to be done is to construct collision-free cubic spirals for each turn around a circular vertex in the C–space. Recall that the edges in $\mathcal{M}$ are made up of directed tangent edges and directed circular arcs, and the vertices are made up of directed tangent contacts. Two directed tangent contacts lying on the same circular vertex with the same orientations (+ or −) determines a turn. If a collision-free cubic spiral can be constructed for a turn, it replaces the corresponding directed circular arc in $\mathcal{E}$ of $\mathcal{M}$. If it cannot be found, the corresponding directed circular arc is removed from $\mathcal{E}$, and the two vertices are disconnected. The resulting smoothed-out visibility net $\mathcal{M}$ is called a smooth $\mathcal{M}$.

Let α denote the angle of a turn, and β be the partially $\dot{\kappa}_{max}$–admissible angle for the circle with radius $r_\beta = r_{min} + \Delta r$. The following technique, which has largely been discussed in the last subsection, can be used to find a collision-free cubic spiral for each turn:

1. If $\alpha \geq \beta$, BR_1 or BR_2 for the cubic spiral can be constructed directly to check

for collisions. If the bounding region collide with obstacles, project the collision points onto the inscribing circle (the circular vertex), and construct a sequence of sub cubic spirals based on these constrained points. If any of these sub cubic spirals has $\alpha < \beta$, report failure. Otherwise check the union of their bounding regions for collision. If it is collision free, report success. Otherwise, report failure.

2. If $\alpha < \beta$, find another inscribing circle with chord length d_β. Follow the procedures in last step to check for collisions against the bounding region of the cubic spiral.

5. Time Complexity Analysis

Assuming there are n obstacle edges and $O(n)$ obstacle corners in W. There are all together $O(n^2)$ vertices in the visibility net. Checking the bounding region of each vertex against collision takes $O(n)$ time. Therefore, it takes $O(n^3)$ time to smooth out the visibility net.

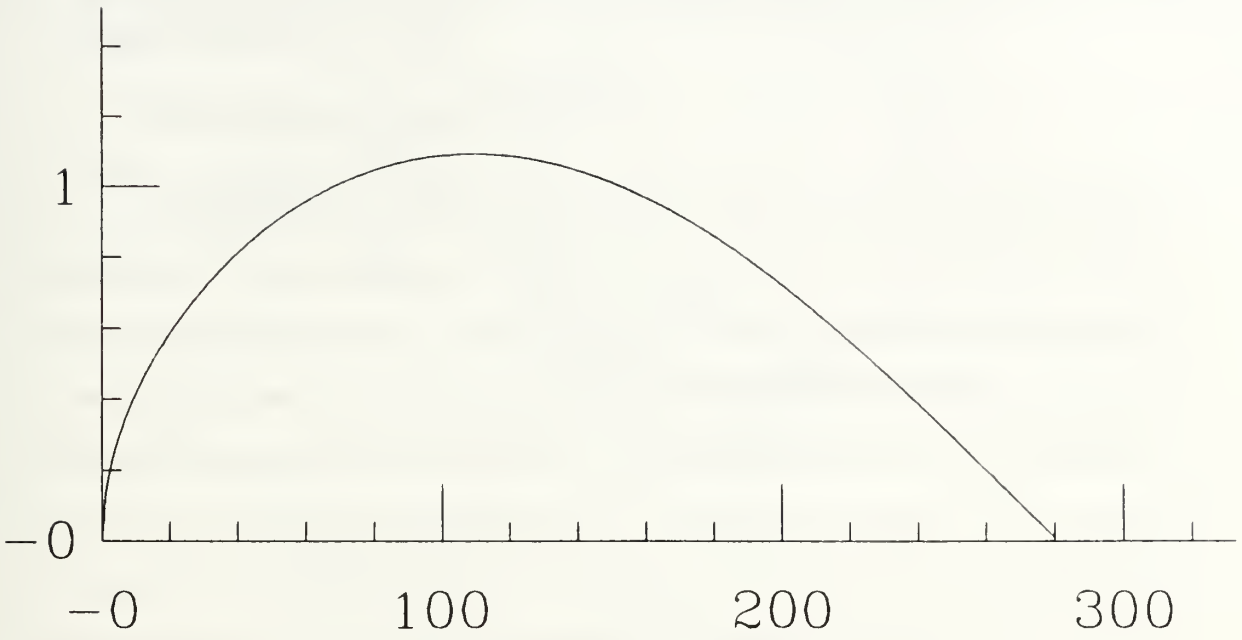


Figure 5.3: $F_4(\alpha)$ Plotted Against α in Steps of 1°

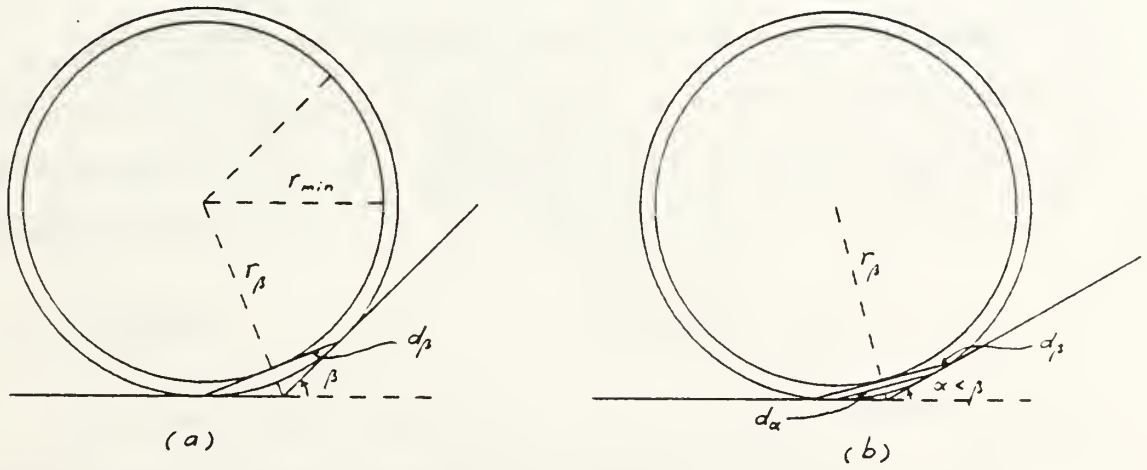


Figure 5.4: Finding Another Circle with $chord_{beta}$

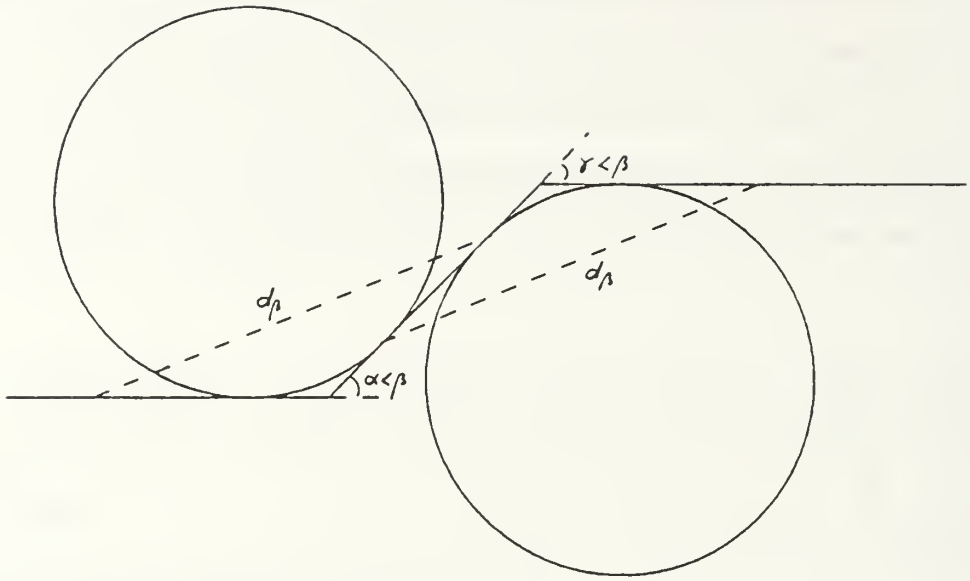


Figure 5.5: Two Very Close Turns of Angles $< \beta$

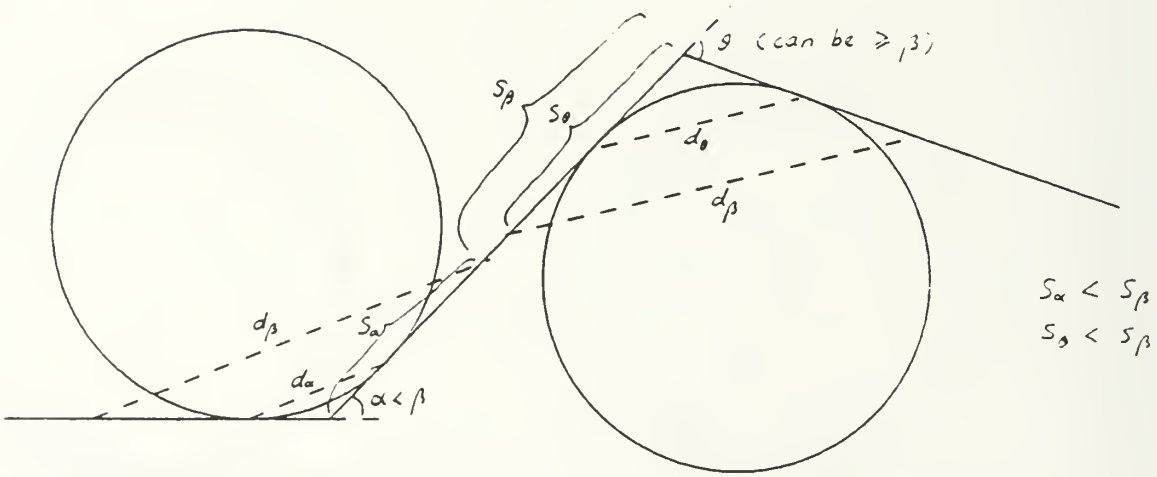


Figure 5.6: Making Room for Two β–Chords

C. SMOOTHING TECHNIQUE FOR THE NOMINAL CONFIGURATION NET

1. The Smoothing Technique

Recall that in a nominal configuration net $\mathcal{M}$, the vertices are the nominal configurations, and the edges are the logical connections. The smoothing technique in this case is simply to replace each logical edge by a physical cubic spiral connecting the vertices, followed by checking the cubic spiral against the kinematic constraints and possible collision with the obstacles. If the checking fails, remove the edge from $\mathcal{M}$. The only complication of this technique is when two adjacent vertices are not a symmetric pair. Constructing a cubic spiral for a non-symmetric pair involves the computation of a *split configuration* in between, and the construction of two sub cubic spirals. This procedure is quite straight forward for a parallel non-symmetric pair, but is non-trivial for a non-parallel non-symmetric pair. The readers are referred to [Ref. 12] for a good description on non-symmetric pairs.

2. Time Complexity Analysis

Assuming there are n obstacle edges in W. There are $O(n)$ vertices and edges in the nominal configuration net. Checking the bounding region of each edge against collision takes $O(n)$ time. Therefore, it takes $O(n^2)$ time to smooth out the nominal configuration net.

3. Critique

While the nominal configuration net can be elegantly smoothed in some cases (see Figure 5.7), it fails in some other carefully chosen examples (see Figure 5.8). The nominal configuration net method is a very good example of an intuitively appealing method which quickly reveals its limitation under scrutiny.

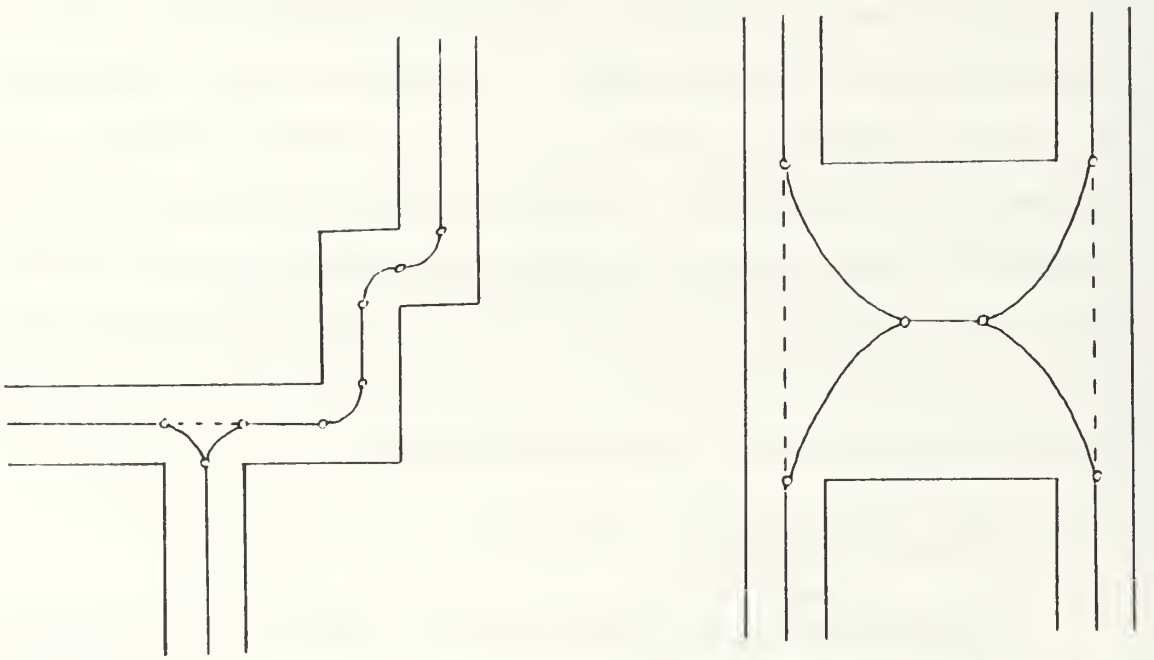


Figure 5.7: Working Examples of the Nominal Configuration Net Method

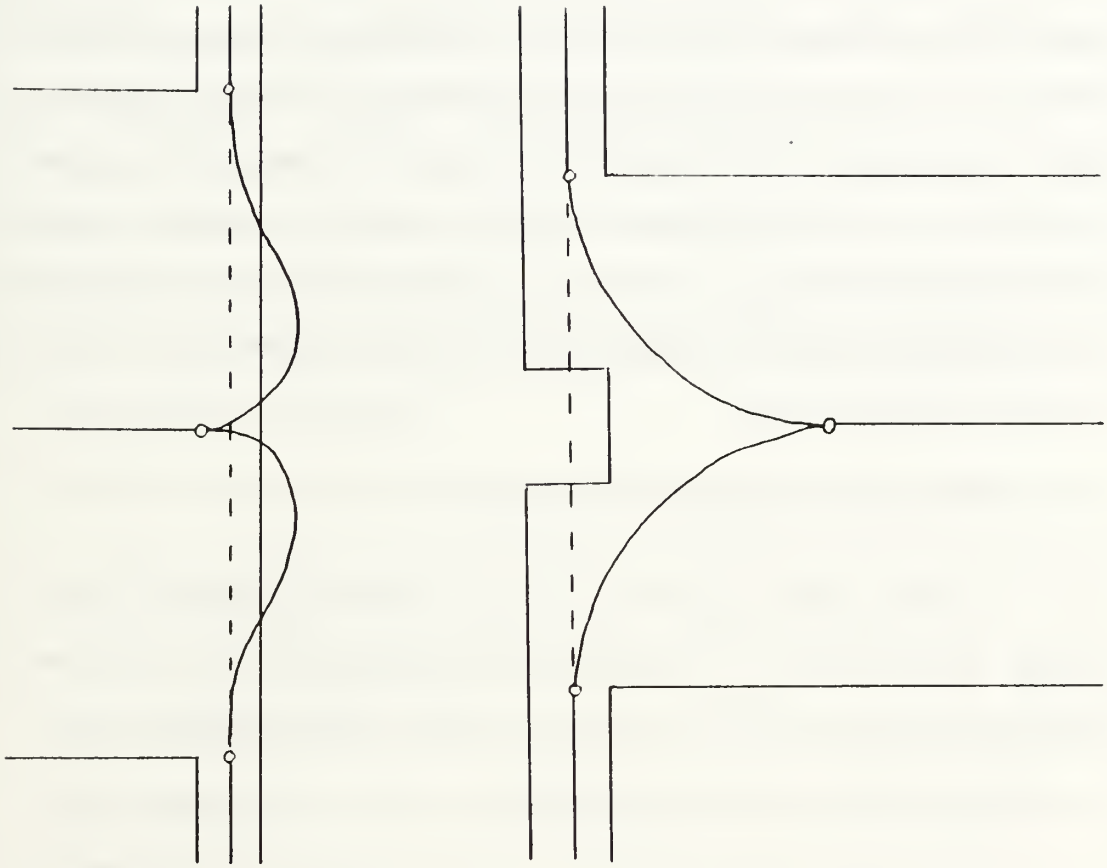


Figure 5.8: Counter Examples of the Nominal Configuration Net Method

D. SMOOTHING TECHNIQUE FOR THE SPINE NET

1. Background

The essence of this technique is to construct collision-free cubic spirals for each directed turn (vertex) in $\mathcal{V}$ of $\mathcal{M}$. Out of the eight directed turns at each junction, two each share the same cubic spiral in different directions (see Figure 3.7). Therefore, there are at most four distinct cubic spirals at each turn. A pair of directed turns in opposite directions is simply called a "turn". If it is not possible to construct a collision-free cubic spiral for a turn under the kinematic constraints, remove the corresponding vertex from $\mathcal{V}$. As mentioned in Chapter IV, there is no closed form of cubic spirals for geometric reasoning. Collisions against obstacles will be checked using their bounding regions.

2. The Smoothing Technique

A turn ($< 180°$) is defined by a pair of intersecting directed spines. Let the angle of the turn be α. A constrained and collision-free cubic spiral can be constructed for each turn in (or to determine failure) with the following steps:

1. The first step is to find the subset of the α–deflection cubic spiral family that meets the κ_{max} and $\dot{\kappa}_{max}$ constraints. From proposition IV.5 and equation 4.13, this subset of cubic spirals are those with size d such that,

$$d \geq d_{min} = \max\left(\frac{3\alpha D(\alpha)}{2\kappa_{max}}, D(\alpha)\sqrt{\frac{6\alpha}{\dot{\kappa}_{max}}}\right)$$

2. The second step is to find the corresponding radii of the inscribing circles of this subset of cubic spirals. This step is necessary for the construction of the bounding regions (be it BR_1 or BR_2). Let r be the radius of the inscribing

circle of the bounding region. From equation 4.13 and equation 4.8, we have,

$$r \geq r_{min} = \max\left(\frac{3\alpha D(\alpha)}{4\kappa_{max}\sin(\alpha/2)}, \frac{D(\alpha)}{2\sin(\alpha/2)}\sqrt{\frac{6\alpha}{\dot{\kappa}_{max}}}\right) \tag{5.3}$$

Note that the centers of the inscribing circles always lie on the bisector of the spines making the turn.

3. The third step is to find the intervals of r that are collision-free. In this step, we represent the disc robot $\mathcal{D}$ as a point robot in its C–space. A radius r is said to be critical if it touches some C–obstacles. If there are m obstacle edges and n obstacle vertices, then there are at most $m + n$ critical radii. A critical interval is defined as the interval between two successive critical radii. Wilfong [Ref. 7] shows that all radii in a critical interval are either collision free or all collide with some obstacles.

   Note that a turn must be completed within the two intersecting directed spines. It is given that the angle of the turn is $\alpha(< 180°)$. Let the two intersecting directed spines be e_1 and e_2. Let lg_1 be the length of the portion of e_1 before the turn, and lg_2 the length of the portion of e_2 after the turn. Without loss of generality, assume $lg_1 \leq lg_2$. Let r_{max} be the maximum radius that the inscribing circle of the cubic spiral turn can assume. From geometry (see Figure 5.9), we have,

$$r_{max} = lg_1 \tan(90° - \frac{\alpha}{2}) \tag{5.4}$$

   A further restriction on the collision-free intervals of r is thus,

$$r_{min} \leq r \leq r_{max}$$

4. The fourth step is to check BR_1 against collision. Suppose $[r_1, r_2], [r_3, r_4], \ldots, [r_{n-1}, r_n]$ are the collision-free critical intervals (see Figure 5.10) with

$$r_{min} \leq r_1 \leq r_2 \leq \cdots \leq r_n \leq r_{max}$$

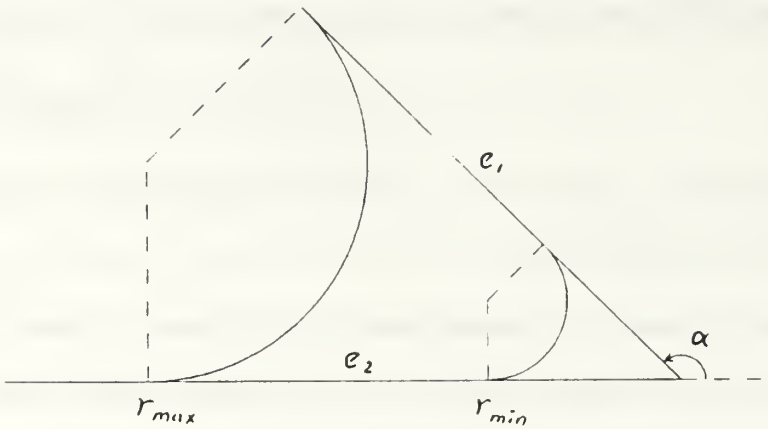


Figure 5.9: Computing r_{max} for a turn

Check only the BR_1 bounded by the circular arc of radius r_1 and the two spines e_1 and e_2 for collision. If BR_1 is collision-free, return success. Otherwise, proceed to the next step. Note that this step always return the smallest possible inscribing circle for the cubic spiral of a turn.

5. The fifth step is to check BR_2 against collision. For each collision-free critical interval $[r_k, r_{k+1}]$, construct a corresponding BR_2 bounded by the circular arc of radius r_{k+1}, the two spines e_1 and e_2, and the tangential cap[1] (see Figure 5.11). Check against each BR_2 for collision. If one of the BR_2 is collision-free, return success. Otherwise, proceed to the next step. Note that this step always return the largest r of the smallest possible critical interval for the cubic spiral of a turn.

6. The last step is to find a sequence of sub cubic spirals (see proposition IV.11), each with deflection $\alpha \geq \beta$, which circumscibed each critical radius r_k. Check the union of their bounding regions against collision. If one of the bounding

[1]There is a conjecture that the two circular arcs of radius r_k and r_{k+1} and the two spines can form a bounding region BR_3 for a cubic spiral if r_{k+1} is suitably larger than r_k. If this can be proved, then we can always check BR_3 (which is trivial) before BR_2.

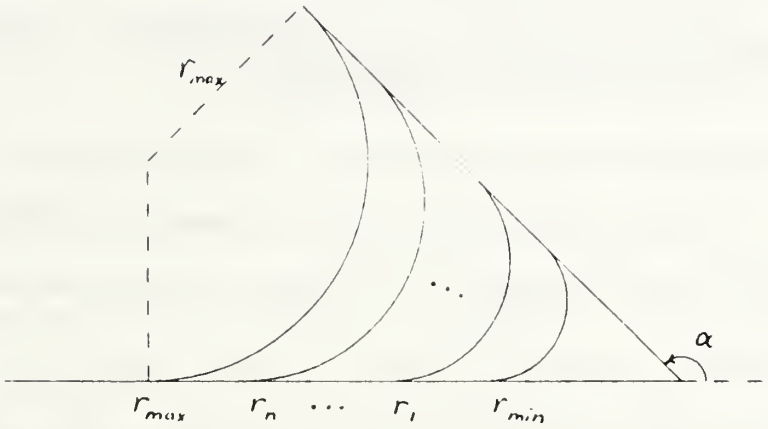


Figure 5.10: Collision-Free Critical Intervals of a Turn

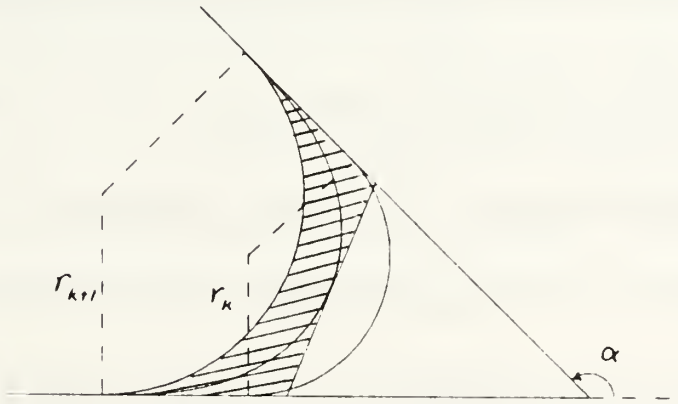


Figure 5.11: Constructing BR_2 for a Critical Interval

region union is collision-free, return success. Otherwise, return failure. Note that this step offers the closest approximations for the cubic spirals.

3. Time Complexity Analysis

1. *Constructing the Convolution Diagram.* Assuming there are n obstacle edges in W. The convolution diagram can be constructed in $O(n^2 \log n)$ time using simple ray sweep technique, or in $O(n \log n)$ time using Voronoi diagram.

2. *Smoothing out the Spine Net.* There are $O(m^2)$ vertices in the spine net. For each vertex (directed turn), there are $O(n)$ critical radii which can be computed in $O(n)$ time. Sorting them in increasing order starting at the smallest critical radius to determine the collision-free critical intervals takes $O(n \log n)$ time. For each critical interval of a directed turn, checking the bounding region against the C–obstacles takes $O(n)$ time. Therefore, it takes a total of $O(m^2 n^2)$ time to smooth out the spine net.

Conclusion Smoothing out the spine net in the local smooth path planning layer takes $O(m^2 n^2)$ time, where $m = O(n^2)$ in the case of freeway method, or $m = O(n)$ in the case of sensor-oriented method.

4. Modifications to the Data Structure

In this layer, two modifications are made to the spine net $\mathcal{M}$:

1. If a collision-free cubic spiral cannot be constructed for a turn under the kinematic constraints, the two corresponding directed turns (vertices) are removed from $\mathcal{V}$. All edges incident with these two vertices are also removed from $\mathcal{E}$.

2. The exit point and the entry point of a directed turn (vertex) are updated according to those of the corresponding cubic spiral turn.

VI. END PORTIONS PATH PLANNING LAYER

A. OVERVIEW

The end portions path planning layer is the third layer in the layered motion planning approach proposed in this thesis. Given S and G, the goal of this layer is to find some candidates of intermediate S_0 (can be coincident with S) and intermediate G_0 (can be coincident with G) such that:

1. It is possible to plan a feasible path from S to S_0.
2. It is possible to merge S_0 into the smooth $\mathcal{M}$ using only cubic spirals.
3. It is possible to diverge from the smooth $\mathcal{M}$ into G_0 using only cubic spirals.
4. It is possible to plan a feasible path from G_0 to G.

At the end of this layer, a number of candidate intermediate configurations of S_0 and G_0 would have been found. Given W, this layer is a per query step that needs to be carried out for each on-line queries of motion planning. A query is a request to plan a smooth path for an initial configuration S and a final configuration G in $free(W)$.

Section B. of this chapter defines the concepts and terms used in this chapter. Section C. considers the basic problem of maneuvering a point configuration into a directed line segment using circular arcs of radius $R_{min} = 1/\kappa_{max}$ and tangent line segments. Section D. considers the inverted problem of maneuvering from a directed line segment into a point configuration. Section E. modifies the basic and the inverted problems into using cubic spirals for the last maneuver subject to κ_{max}

and $\dot{\kappa}_{max}$ constraints. Section F. makes use of the techniques developed in earlier sections to complete the end portions path planning problem of the spine net. Lastly, section G. analyses the time complexity of the end portions path planning technique.

Important Notes:

1. The end portion path planning technique developed in this chapter is targeted only for spine nets. It can be applied to the visibility net, but the result is less satisfactory. This is because the technique in this chapter looks for spines for merging, while in the case of visibility net, another technique which looks for circular vertex for merging would be more optimal. This technique is not applicable to the nominal configuration net at all. Time permitted, the author may investigate into the techniques for the visibility and the nominal configuration nets in future.

2. In this chapter, the dic robot $\mathcal{D}$ is always represented as a point configuration p in its configuration space (C–space).

B. DEFINITIONS

Given a directed line segment e, denote the directed line extended from e as $l(e)$. A circle can lie in three positions relative to $l(e)$, and hence to e: $\mathcal{L}$ *(left)*, $\mathcal{R}$ *(right)*, and $\mathcal{O}$ *(on)*. See Figure 6.1.

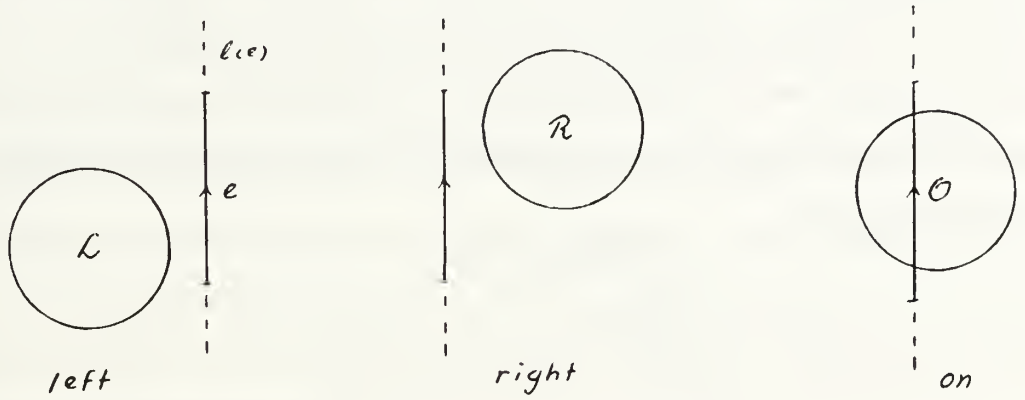


Figure 6.1: Relative Positions of Circles Against a Directed Line Segment

There are two *orientations* defined for a circle: *L mode (CCW)* and *R mode (CW)*. Likewise applies to a circular arc.

Combining relative positions and orientations, there are six possible *relative configurations* a circle can have against a directed line segment: $\mathcal{L}L, \mathcal{L}R, \mathcal{R}L, \mathcal{R}R, \mathcal{O}L, and \mathcal{O}R$, where the first letter represents the relative position and the second letter the orientation. See Figure 6.2.

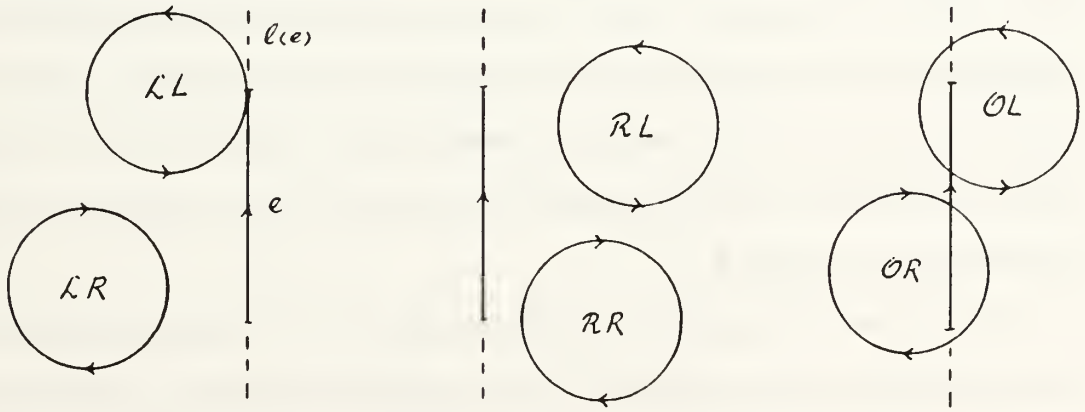


Figure 6.2: Relative Configurations of Circles Against a Directed Line Segment

A circle with curvature κ_{max} has radius $1/\kappa_{max}$ denoted by R_{min}. We call the circle an R_{min}–circle. Likewise applies to an R_{min}–arc.

Given a point robot p (represented by its configuration), two R_{min}–circles tangential to p can be defined: one is an L–circle, the other an R–circle. If the circles are obstructed by obstacles, L–arc and R–arc are defined instead. These two tangential arcs can be further divided into four maneuvering arcs: forward L–arc, backward L–arc, forward R–arc, and backward R–arc. The concept of forward and backward is based on the maneuvering direction (forward or backing-up) of p on the maneuvering arc. See Figure 6.3.

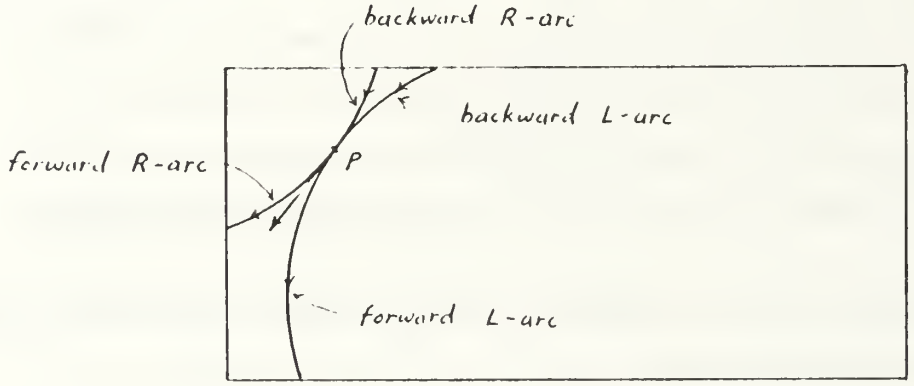


Figure 6.3: Four Types of Maneuvering Arcs Tangential to p

The *extremity* of a maneuvering arc is where p maneuvers to the other end of the maneuvering arc and hits the obstacle wall. From the extremity, p can turn the front wheels to the opposite maximum steering angle and reverse its velocity, thus tracing out a new maneuvering arc with opposite mode and opposite maneuvering directions. Note that two consecutive maneuvering arcs share a common vertical tangent. For example, from the extremity of a forward L–arc, a backward R–arc can be generated. See Figure 6.4.

The number of *maneuvers* of a path is defined as the number of maneuvering arcs involved in the path. Intuitively, the less number of maneuvers, the better the solution path.

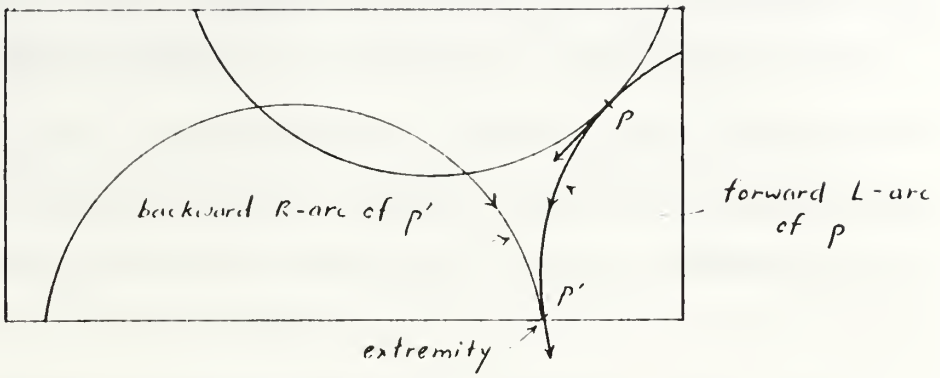


Figure 6.4: Tracing Out a New Maneuvering Arc from an Extremity

When p is maneuvered from an initial configuration S into a directed line segment e, p (or S) is said to *merge* into e. When p is maneuvered from e into a final configuration G, p (or G) is said to *diverge* out of e. These terminologies will be used in subsequent discussions.

C. MERGING A CONFIGURATION INTO A DIRECTED LINE SEGMENT USING R_{min}–CIRCULAR–ARCS

This section attempts to solve only the basic problem of merging p into e using only circular arcs and tangent segments. That is, p is subject to κ_{max} constraint only, and reversals are allowed. A predefined cap of k is placed on the maximum number of maneuvers allowed in a solution path.

In [Ref. 16], P. Jacobs and J. Canny show that it is enough to consider paths which consist of circular arc segments of maximum curvature (the R_{min}–arcs) and common tangents in path planning subject to κ_{max} constraint. The principle will be adopted in this section.

1. Without Considering Obstacles

Consider connecting two configurations S and G in free space using only R_{min}–arcs and common tangents, there are at most four possible solution paths. See Figure 6.5.

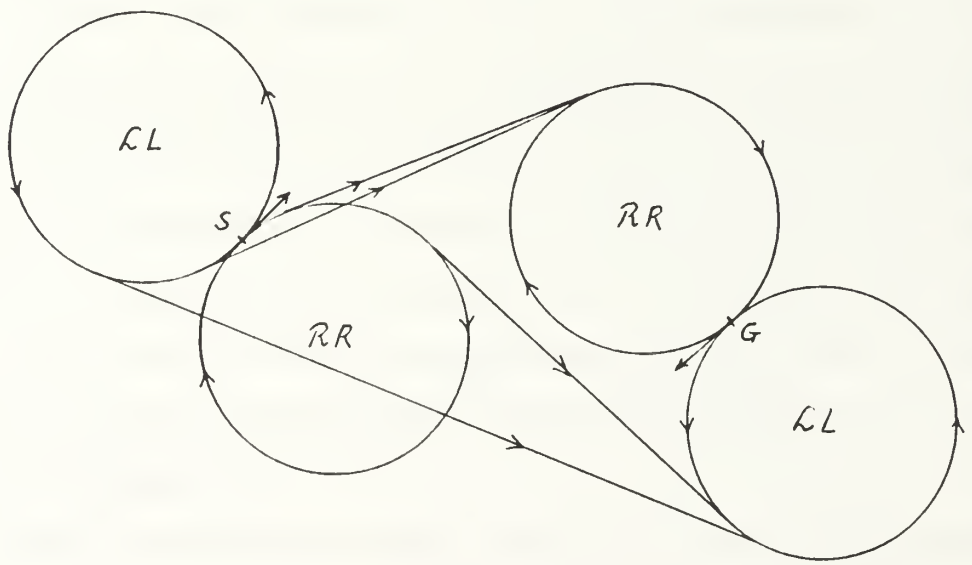
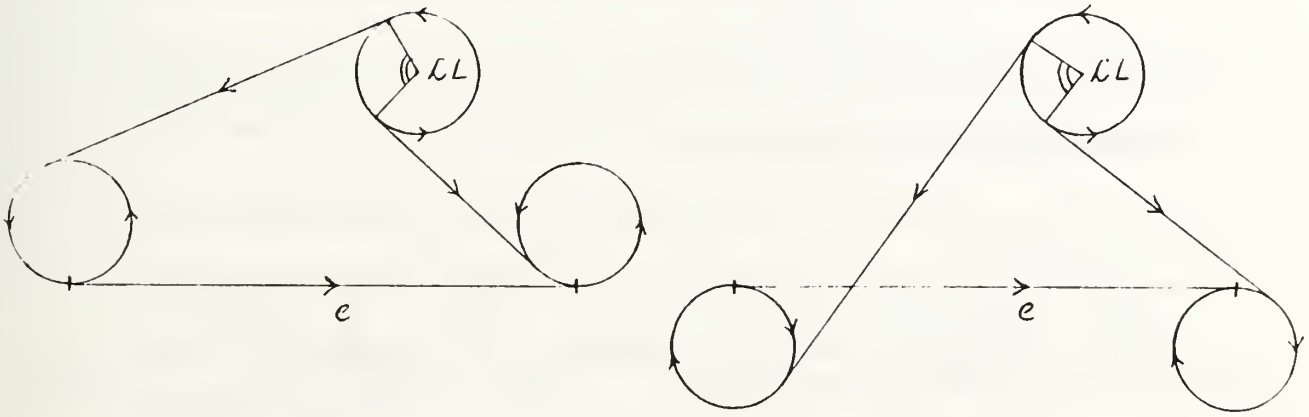


Figure 6.5: Solution Paths Connecting Two Configurations

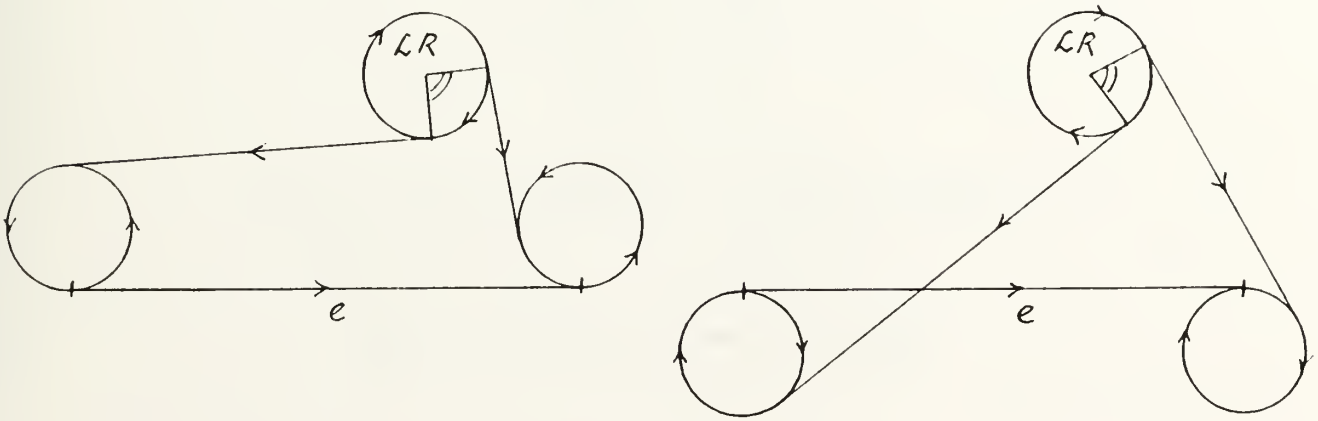
In contrast, there are infinitely many solution paths for merging a configuration p into a directed line segment e. If the L and R circles of p are considered separately, this problem can be divided into the following cases according to the relative configurations of a directed circle against e. The solution ranges are shown in the figures, where the distance from the center of a circle to e is denoted as z.

Cases of Solution Ranges:

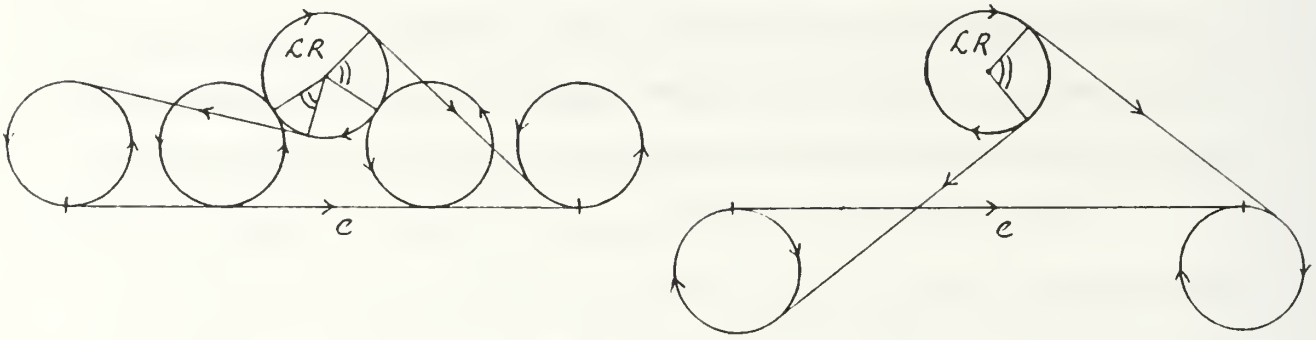
1. Mode $\mathcal{L}L/\mathcal{R}R$ and $z \geq R_{min}$



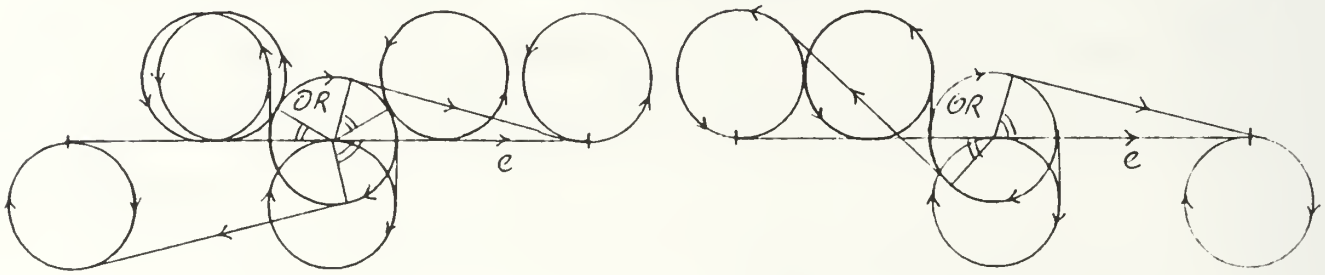
2. Mode $\mathcal{L}R/\mathcal{R}L$ and $z \geq 3R_{min}$



3. Mode $\mathcal{L}R/\mathcal{R}L$ and $3R_{min} > z \geq R_{min}$



4. Mode $\mathcal{O}L/\mathcal{O}R$ and $R_{min} > z$



2. Limiting the Solution Ranges

The solution ranges presented in the last subsection can be further limited based on the following empirical arguments (mainly targeted for the end portions planning in a spine net to be discussed shortly):

- Only a finite number of solution paths can be considered for purposes such as collision checking or entry point selections. Therefore, a finite subset of the infinite solution range with interesting properties should be sufficient.

- Solution paths which cross over to the other side of e are not interesting if there are plenty opportunities in choosing solution paths on the same side.

- Only "forward mergings" are of interest, where the entry point of merging is in front of the projection of the L or R circles on e. Though a restriction to directed line segments, it is generally adequate for p to merge into undirected line segments at any entry point. See Figure 6.6.

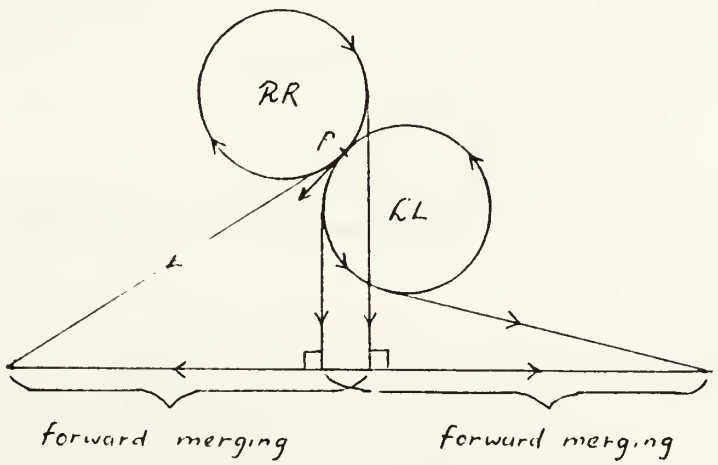
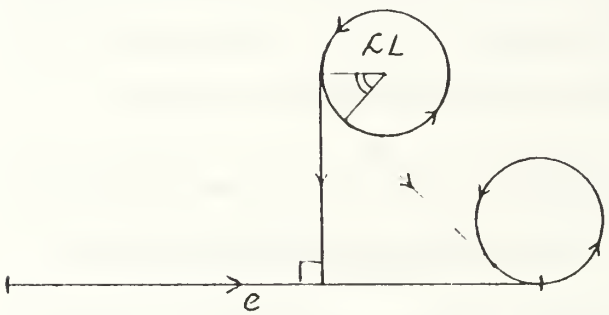


Figure 6.6: Adequacy of Forward Mergings

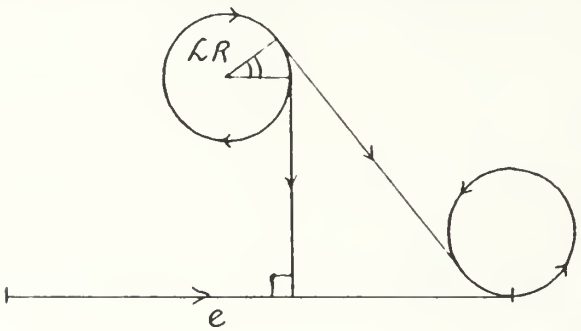
Based on these empirical arguments, the solution ranges of the last subsection are limited as follows. Note that cases 2 and 3 of the last subsection are collapsed into a single case.

Cases of Limited Solution Ranges:

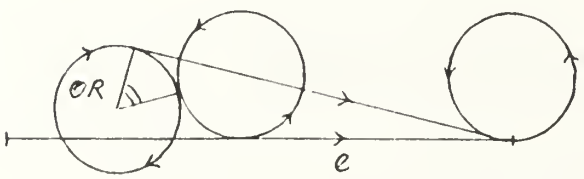
1. Mode $\mathcal{L}L/\mathcal{R}R$ and $z \geq R_{min}$



2. Mode $\mathcal{L}R/\mathcal{R}L$ and $z \geq R_{min}$



3. Mode $\mathcal{O}L/\mathcal{O}R$ and $R_{min} > z$



3. Considering Obstacles

Existence of obstacles poses the following problems:

- The L and R circles of p may be obstructed by some obstacles in C-space, thus reducing to L and R arcs.
- The directed line segment e may be partially visible or be completely blocked by some obstacles from the L and R circles or arcs of p.

Fortunately, it is easy to see that so long as the limited solution range is non-empty for an interval of e visible to a tangential circle or maneuvering arc of p, all solution paths in the limited solution range are guaranteed collision-free. See Figure 6.7.

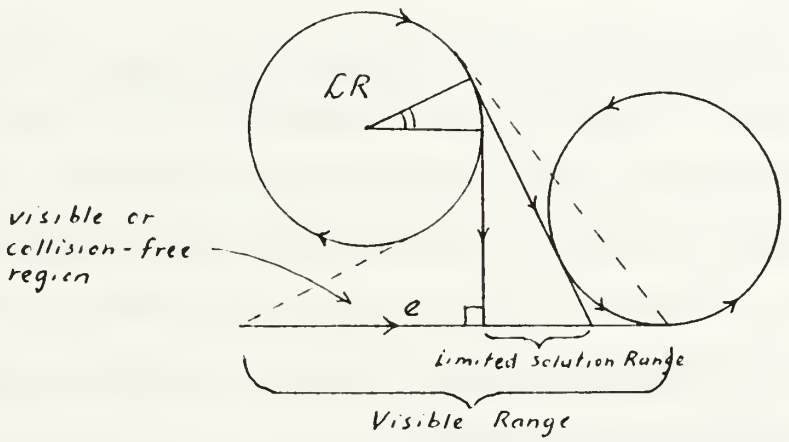


Figure 6.7: The Collision-Free Limited Solution Range

Proposition 6.1 *The limited solution range is always a subset of the visible range of a configuration p against a directed line segment e. At the least, there is always a difference incurred at the front end of the visible interval of e where the furthest entry point of p (with the smallest merging deflection angle α) falls short of s_α as shown in Figure 6.7.*

Proposition 6.2 *All solution paths in the limited solution range are guaranteed collision-free.*

With these considerations in mind, the following empirical procedure can be employed for merging a configuration p into a directed line segment e amidst obstacles. Note that the procedure is obviously very much incomplete for merging a given p into a single directed line segment e. But in later sections, it can be seen that the procedure is quite sufficient for merging p into any of the spines in a spine net.

Procedure Merge:

1. **Step 1** Construct the L and R circles of p, and check them against obstacles. If none of them is collision free, goto step 5.

2. **Step 2** If e is not visible to either of the collision-free circles of p, report failure and return.

3. **Step 3** For each of the L and R circles of p that sees e, do the following: If more than one intervals of e are visible from the circle because of multiple obstacles standing in between p and e, choose the interval that "contains" or "is the nearest in front of" the projection of the circle on e. Make this interval the new e (see Figure 6.8). Otherwise, there is only one interval of e visible to the circle; make this interval the new e. Determine the limited solution range of the circle against e (can be empty).

4. **Step 4** If both the limited solution ranges of the L and R circles of p are empty, report failure and return. Else, goto step 10.

5. **Step 5** Break the L–arc of p into forward L–arc and backward L–arc. Do the same thing to the R–arc. There are now four maneuvering arcs of p under consideration. Let i denote the number of maneuvers so far. Set $i = 0$.

6. **Step 6** For each of the forward L and R arcs and the backward L and R arcs, do the following:

(a) **Step 6.1** If e is not visible, set the limited solution range of the arc to empty, and continue with the next iteration loop of *step 6.*

(b) **Step 6.2** If more than one intervals of e are visible from the arc because of multiple obstacles standing in between p and e, choose the interval that "contains" or "is the nearest in front of" the projection of the arc on e. Make this interval the new e. Otherwise, there is only one interval of e visible to the arc; make this interval the new e. Determine the limited solution range of the arc against e (can be empty).

7. **Step 7** If any of the limited solution ranges of the four arcs are non-empty, goto step 10.

8. **Step 8** Set $i = i + 1$. If $i > k$, where k is the maximum number of maneuvers allowed (a predetermined parameter), then report failure and return.

9. **Step 9** From the extremities of each of the four arcs, generate four new collision-free maneuvering arcs which are opposite in both the modes and directions. Goto step 6.

10. **Step 10** At this step, a number of candidate limited solution ranges from each of the collision-free circles or arcs of p against visible e have been obtained. For each of the limited solution ranges, choose a candidate solution path with the largest merging deflection angle, that is, closest to the projection point of the circles or arcs of p on e. From Proposition VI.2 and VI.1, these candidate solution paths are guaranteed to be collision-free. Pick one of them with the least number of backward arcs as the solution path. If there is a tie, pick the one with the least number of arcs. If there is a tie again, pick one randomly. Report success and return the solution path.

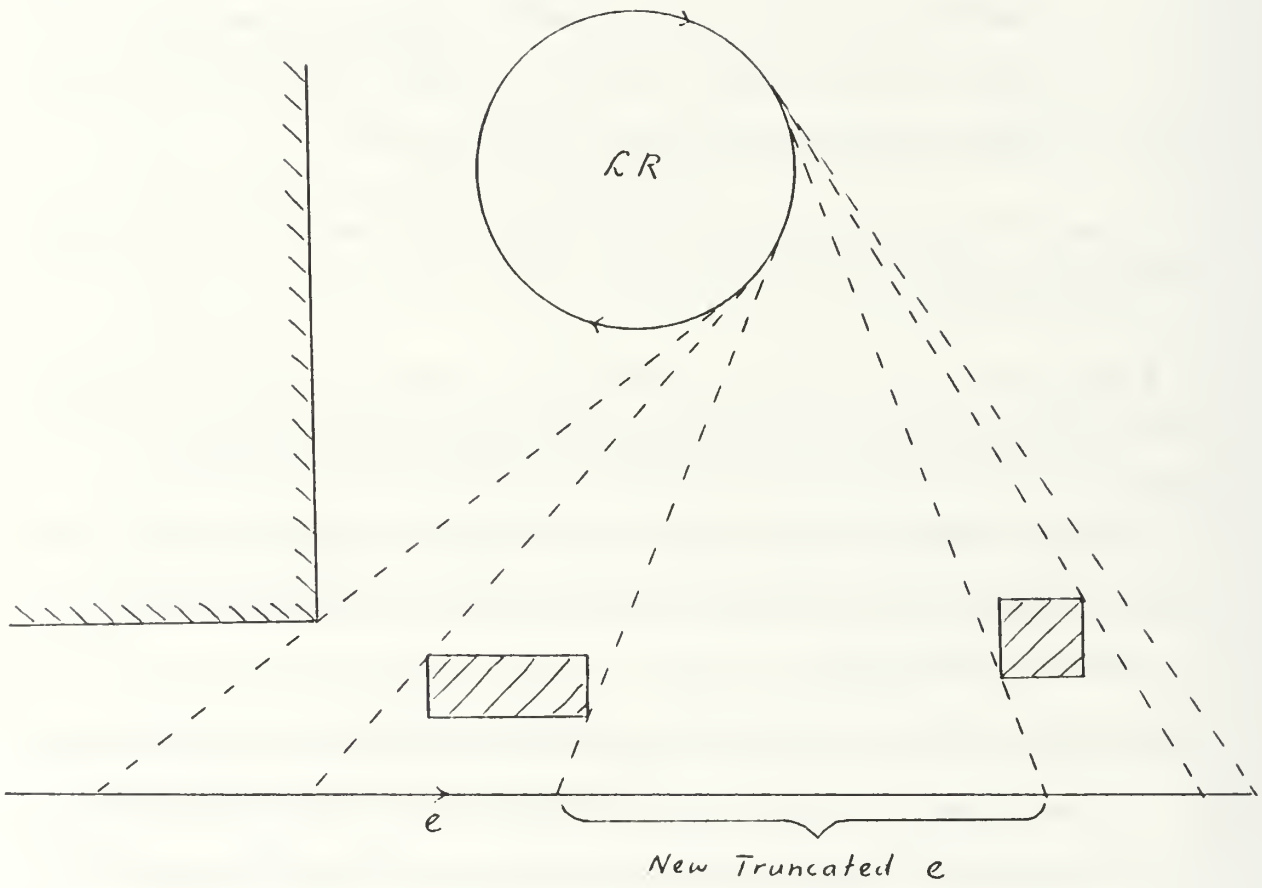


Figure 6.8: Partially Visible e and the New Truncated e

D. DIVERGING A CONFIGURATION OUT OF A DIRECTED LINE SEGMENT USING R_{min}–CIRCULAR–ARCS

Because of the symmetry of the solution path between the merging problem and the diverging problem, it is not necessary to design another procedure for solving the diverging problem. Rather, we can invert the diverging problem to the merging problem of the last section, solve it, then invert the solution back for the diverging problem.

More specifically, the procedure for solving the diverging problem is as follows:

Procedure Diverge:

1. **Step 1** Invert the orientation of p and e.

2. **Step 2** Solve the problem of merging the inverted p into the inverted e using *Procedure Merge* of the last section.

3. **Step 3** If no inverted merging solution can be found, report failure and return. Else, invert the order and orientation of all path segments (and maneuvering directions) of the inverted merging path to obtain the desired solution path for diverging. Report success and return.

E. MANEUVERING BETWEEN A CONFIGURATION AND A DIRECTED LINE SEGMENT USING CUBIC SPIRALS

Recall that one of the major consideration of using cubic spirals instead of circular arcs is to allow non-stop forward-moving motions for p in making smooth turns during the middle portion of a motion path. However, at the end portions (the initial and the final portions), stops and reversals are allowed, and only feasible paths are called for. Therefore, in end portions path planning:

- It is not necessary to replace circular arcs by cubic spirals, because p will have to halt at the extremities anyway. Circular arcs will suffice.

- The only exceptions are: cubic spiral is used for the last maneuver of p merging into e, and for the first maneuver of p diverging out of e. In fact, these two places defines the intermediate S_0 and G_0 configurations called for in the problem statement of this thesis. They are also the demarcation points between the middle portion and the end portions of a solution path.

In the discussions that follows, only κ_{max} constraint is considered at first for merging p into e in the initial phase using cubic spirals. Then an additional $\dot{\kappa}_{max}$ constraint is imposed on the cubic spiral. Lastly, the problem of diverging p out of e in the final phase using cubic spirals subject to both constraints is considered. This section makes heavy references to the last two sections, because paths consisting of circular arcs are the stepping stones towards constructing those of cubic spirals. Obstacles are considered throughout this section.

1. Merging p into e Using Cubic Spirals Subject to κ_{max} Constraint Only

Procedure Merge as described in subsection 3. can be modified as follows to find a solution path using cubic spirals for the last maneuver of p into e in the initial portion planning:

Modifications to Procedure Merge:

1. Instead of using R_{min}–circles in *Procedure Merge*, use r_{min}–circles, where

$$r_{min} = \frac{3}{2\kappa_{max}}$$

   This is to ensure full κ_{max}–admissibility for the circumscribing cubic spirals.[1] All steps in *Procedure Merge* are affected by this modification.

2. For the last maneuver in the initial portion planning, if a circular turn is required on the maneuvering arc of p in the forward direction, it must be replaceable by a collision-free circumscribing cubic spiral (see Figure 6.9). Add an additional *step 6.3* to *Procedure Merge* as follows:

   - **Step 6.3** Let q denote the configuration on the last maneuvering arc of p with the largest merging deflection angle in the limited solution range. If $p = q$, or the last maneuvering arc is a backward arc, then make p a candidate S_0 called for in the problem statement and continue with the next iteration loop of *step 6*. Else, construct the bounding region (BR_1 or BR_2) of the cubic spiral circumscribing on p and q. If this bounding region is collision-free, make p a candidate S_0. Otherwise, make q the candidate S_0.

[1] Although the R_{min}–circle can still be used for the circular maneuvering arcs, such minor optimization is omitted for the sake of simplicity.

3. For the last maneuver in the initial portion planning, the circular arc merging into e has to be replaced by a cubic spiral. It can be easily seen that this circular arc and its two tangents (one of which lies on e) form a bounding region BR_1 of the cubic spiral circumscribing the circular arc. Since BR_1 is contained in the visible region between the maneuvering arc of p and e, BR_1 (and hence the cubic spiral) are clearly collision- free (see Figure 6.9). In *step 10* of *Procedure Merge*, simply replaces all circular arcs in the last maneuvers by their circumscribing cubic spirals.

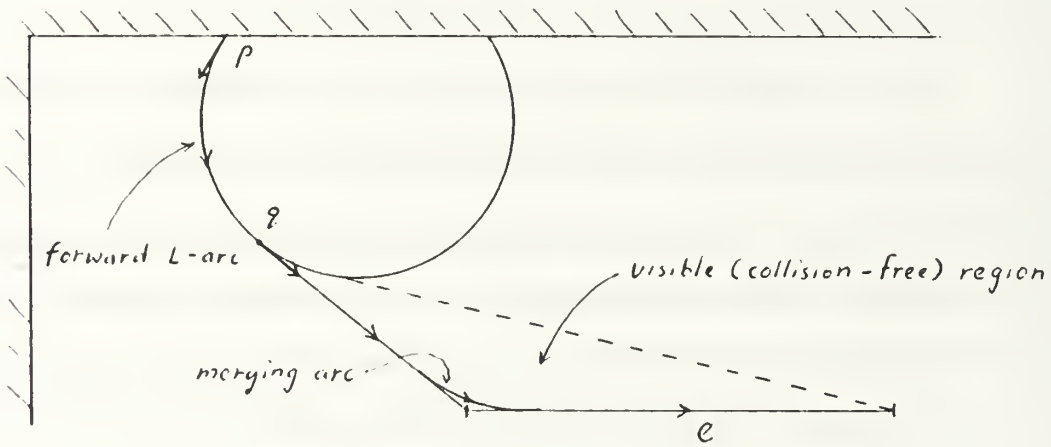


Figure 6.9: Final Maneuver in the Initial Phase Using Cubic Spiral Turns

2. Merging p into e Using Cubic Spirals Subject to an Additional $\dot{\kappa}_{max}$ Constraint

Assuming that β is the partially $\dot{\kappa}_{max}$–admissible angle for the r_{min}–circle. *Procedure Merge* has to be further modified as follows to cater for an additional $\dot{\kappa}_{max}$ constraint:

Further Modifications to Procedure Merge:

1. The furthest entry point of p into e is restricted to have merging deflection angle $\alpha \geq \beta$ (see Figure 6.10). Although this constraint sometimes further reduces the limited solution range, it guarantees that the problem of partial $\dot{\kappa}_{max}$–admissibility can be avoided all together for the circular arc merging into e during the final maneuver. *Step 3* and *step 6.2* of *Procedure Merge*, which are responsible for constructing the limited solution range, are affected by this modification.

2. Modify *step 6.3* and add additional *steps 6.4* and *6.5* to *Procedure Merge* as follows:

   - **Step 6.3** Let q denote the configuration on the last maneuvering arc of p with the largest merging deflection angle in the limited solution range. If $p = q$, or the last maneuvering arc is a backward arc, then make p a candidate S_0 called for in the problem statement and continue with the next iteration loop of *step 6*. Else, if the deflection angle from p to q is $\geq \beta$, goto *step 6.5*.

   - **Step 6.4** Let q' denote the configuration on the final maneuvering arc of p with the smallest merging deflection angle ($\geq \beta$) in the limited solution range. If the deflection angle from p to q' is $\geq \beta$, make q' to be q, and goto *step 6.5*. Else, reset the limited solution range of this maneuvering

arc against e to empty, and continue with the next iteration loop of *step 6.*

- **Step 6.5** Construct the bounding region (BR_1 or BR_2) for the cubic spiral circumscribing on p and q. If this bounding region is collision-free, make p a candidate S_0. Otherwise, make q the candidate S_0.

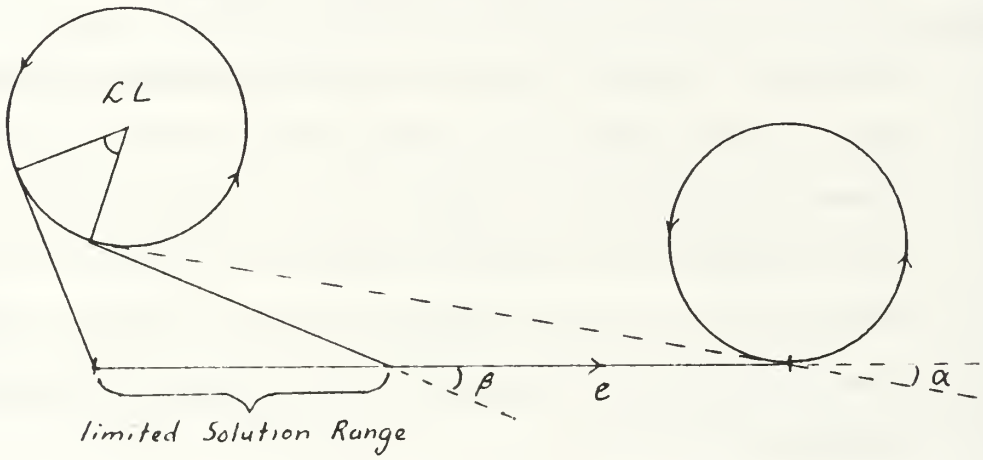


Figure 6.10: The Collision-Free Limited Solution Range subject to $\dot{\kappa}_{max}$ Constraint

3. Diverging p out of e Using Cubic Spirals Subject to Both κ_{max} and $\dot{\kappa}_{max}$ Constraints

As is the case of the circular arc, the cubic spiral is also symmetric. Therefore, *Procedure Diverge* as described in section D. can be followed exactly to solve the problem of diverging p out of e using cubic spirals subject to both the κ_{max} and $\dot{\kappa}_{max}$ Constraints for the final portion planning. Of course, the modified *Procedure Merge* is employed in finding the inverted solution path in *Procedure Diverge.*

F. END PORTIONS PATH PLANNING FOR THE SPINE NET USING CUBIC SPIRALS

This section is a culmination of the techniques developed in the previous sections for end portions path planning for the spine net. The following procedure simply applies *Procedure Merge* and *Procedure Diverge* to S and G respectively against all directed spines in the spine net, thereby generating many candidates of S_0 and G_0.

Procedure End Portions Path Planning

1. **Step 1** For each of the directed spines, do the following:

   Use *Procedure Merge* to check if it is possible to merge S into it. If yes, note down the candidate S_0 and the entry point of this directed spine.

2. **Step 2** If the set of candidate S_0 is empty, report failure and return.

3. **Step 3** For each of the directed spines, do the following:

   Use *Procedure Diverge* to check if it is possible to diverge G out of it. If yes, note down the candidate G_0 and the exit point of this directed spine.

4. **Step 4** If the set of candidate G_0 is empty, report failure and return.

There is a fine point that can be made about *Procedure End Portions Path Planning*. At the moment, the last maneuver into a directed spine during the initial portion planning, and the first maneuver out of a directed spine during the final portion planning, are always forward-moving. In case there is a narrow passage where the only possible last maneuver for S is to reverse out, or the only possible first maneuver at G is to reverse in, *Procedure End Portions Path Planning* will always fail. It is surprisingly easy to handle these cases by adding some more steps into *Procedure End Portions Path Planning* in conjunction with the *Procedure Graph*

Search to be discussed in the next chapter. The necessary changes are sketched as follows without increasing the time complexity of the operations:

1. In case *Procedure End Portions Path Planning* fails to find a candidate S_0 or G_0, reverse the direction of S and/or G accordingly, and apply the procedure again.

2. If a candidate can be found this time, apply *Procedure Graph Search* to obtain an *auxiliary* candidate solution path. Else, report failure.

3. If it is S that is in question, find the first junction on the auxiliary candidate solution path where it is possible for p to turn into an opposite direction. Then it is obvious that p could reverse all the way from S to this point, and resume forward-moving motion from this point on. Note that because of symmetry, it is possible to obtain a forward-moving path by reversing the directions of a backward-moving path.

4. Similar argument applies to the case of G.

This refinement is not included in *Procedure End Portions Path Planning* for a simple reason: this procedure and the overall layered motion planning problem using spine net is incomplete. They are practical because office-like environment is assumed where there are ample leeway for maneuvering. Consideration of narrow passage which requires p to reverse out of S or into G is just not consistent with the assumption, and is only of theoretical interest (although this refinement offers to solve the narrow-passage problem gracefully).

G. TIME COMPLEXITY ANALYSIS

Referring to the data structure of the spine net discussed in Section 4., there are $O(m)$ directed spines in a spine net. In the case of sensor-oriented spine net, $m = O(n)$, where n is the number of obstacle walls in W. In the case of freeway spine net, $m = O(n^2)$.

Both *Procedure Merge* and *Procedure Diverge* take $O(nk)$ time. Hence *Procedure End Portions Path Planning* takes $O(mnk)$ time.

Conclusion *Procedure End Portions Path Planning* takes $O(mnk)$ time.

VII. GRAPH SEARCH LAYER

A. OVERVIEW

The graph search layer is the fourth and the last layer in the layered motion planning approach proposed in this thesis. In this layer, a Dijkstra's search is made for all candidate S_0 and G_0 (generated by the end portions path planning layer) on the smooth $\mathcal{M}$, producing a number of candidate solution paths. Subsequently, an optimal solution path from S to S_0 to G_0 to G is selected out of these candidate solution paths based on a predefined cost function. Note that this solution path is the most optimal only in the context of the layered motion planning approach proposed in this thesis. This solution path is also a qualified solution to the problem statement set forth in this thesis.

Notes:

1. Factors that can be considered in a cost function include:

   - path length
   - path smoothness
   - safety
   - number of turns
   - number of reversals
   - wall following distance

   Actual definition of a cost function is not important in theoretical studies. However, in actual applications, specific cost functions must be defined for each specific case.

2. Although the graph search procedure developed in this chapter is worded specifically for the spine net, the general principle is applicable to other classes of roadmaps as well.

B. PROCEDURE GRAPH SEARCH

Given a number of candidate S_0 and G_0 generated by *Procedure End Portions Path Planning* of the last layer, the following procedure produce an optimal solution path from S to S_0 to G_0 to G based on a predefined cost function.

Procedure Graph Search

1. **Step 1** For each of the candidate G_0, do the following:
   (a) **Step 1.1** Associated with each candidate G_0 are a directed spine e_g that it diverges out of, and the corresponding *exit point* G_1 on e_g. Make G_1 an auxiliary vertex in $\mathcal{M}$ with no outgoing edges.
   (b) **Step 1.2** Compute the path cost from G_1 to G_0 to G according to the predefined cost function. Assign this cost as the initial cost of the auxiliary vertex G_1.
2. **Step 2** For each of the candidate S_0, do the following:
   (a) **Step 2.1** Associated with each candidate S_0 are a directed spine e_s that it merges into, and the corresponding *entry point* S_1 on e_s. Make S_1 an auxiliary vertex in $\mathcal{M}$ with no incoming edges.
   (b) **Step 2.2** Compute the path cost from S to S_0 to S_1 according to the predefined cost function. Assign this cost as the initial cost of the auxiliary vertex S_1.
   (c) **Step 2.3** Make a Dijkstra's Search of $\mathcal{M}$ starting from the auxiliary vertex S_1. Whenever an auxiliary vertex G_1 is reached, a candidate solution path

from S to S_0 to S_1 to G_1 to G_0 to G is found. This candidate solution path is also the least expensive one that passes through the candidate S_0 in this *step 2* loop. Terminate the Dijkstra's Search and continue with the next loop iteration. Note that the Dijkstra's Search must be modified to include the initial costs of S_1 and G_1.

3. **Step 3** If none of the auxiliary vertices G_1 is reached in *step 2*, report failure and return.

4. **Step 4** Of all the auxiliary vertices G_1 reached in *step 2*, pick the candidate solution path with the least cost. This is the most optimal solution path found using the layered motion planning approach proposed in this thesis. It is also a solution to the problem statement of this thesis.

C. TIME COMPLEXITY ANALYSIS

In the case of a spine net $\mathcal{M}$, there are potentially $O(m^2)$ vertices and $O(m^4)$ edges, where m is the number of spines in the spine net. Hence a Dijkstra's Search on $\mathcal{M}$ can be carried out in $O(m^4)$ time.

The end portions path planning layer can generate up to $O(m)$ number of candidate S_0 and G_0. Hence *Procedure Graph Search* takes up a total of $O(m^5)$ time.

In the case of sensor-oriented spine net, $m = O(n)$, where n is the number of obstacle walls in W. In the case of freeway spine net, $m = O(n^2)$.

Conclusion *Procedure Graph Search* takes $O(m^5)$ time.

VIII. IMPLEMENTATION PLAN

A. INTRODUCTION TO YAMABICO-11 ROBOT

Yamabico-11 is a research robot under continuous development by students and faculty at the Naval Postgraduate School. It is a Power Wheeled Steering (PWS) type vehicle, and can be closely approximated by a disc-shaped rigid body robot in its planar applications. Its low level motion control functions allow it to follow circular paths, cubic spiral paths and straight paths (which are just special cases of cubic spiral paths). At present, seven sonar sensors mounted on it are fully functional which allows position and orientation correction using sonar returns.

The targeted operating environment of Yamabico-11 robot is an in-door office environment with mostly rectilinear type of constructions and obstacles. Because of engineering imprecisions, it is imperative for Yamabico-11 robot to perform continuous position and orientation correction when following a preplanned motion path. The quality of the sonar return is best when the robot navigates in parallel with an obstacle wall. Therefore, it is important that the motion planar of the robot plans motion paths that follows obstacle walls in an optimal way.

In this respect, the layered motion planning approach using sensor-oriented method in the global path planning layer is directly applicable to the Yamabico-11 robot.

B. KINEMATIC CONSTRAINTS OF YAMABICO-11 ROBOT

Although Yamabico-11 is a non-holonomic robot (meaning that its heading direction (reference orientation) must be equal to the tangent orientation of the motion path), it can rotate about its center when it is stationary. This allows it to follow paths consisting of only straight line segments with frequent stoppings at each junction. Yamabico-11 is also not subject to any curvature constraint because it is a PWS type vehicle.

However, it would be desirable to plan motion paths in an office-like environment which Yamabico-11 can navigate with non-stopping and forward-moving motion. Such a requirement in effect imposes three artificial constraints, namely the C^2 path class, the κ_{max} and $\dot{\kappa}_{max}$ constraints, on Yamabico-11 depending on its traveling speed.

1. Let the tread of Yamabico-11 robot be $2W$. According to [Ref. 17], if the robot is moving at a speed v, the speed v_l and v_r at the left and right wheels are:

$$v_l = \left(\frac{r - W}{r}\right) v = (1 - W\kappa)v \tag{8.1}$$

$$v_r = \left(\frac{r + W}{r}\right) v = (1 + W\kappa)v \tag{8.2}$$

where κ is the curvature of the motion path.

These equations completely describes the kinematics of PWS vehicles. They show that the speeds of both the left and right wheels are linear functions of κ. Therefore, if $\kappa(s)$ is discontinuous, the vehicle is not able to follow the path unless it stops at those points. If Yamabico-11 robot is to navigate in a non-stop motion, its motion path must have curvature continuity, that is its motion path must be of class C^2. An example is a path consisting of cubic spiral segments.

2. Let m be the mass of Yamabico-11 robot. Assuming constant traveling speed v for Yamabico-11, the centripetal force during a turn is:

$$\frac{mv^2}{r} = mv^2\kappa \tag{8.3}$$

If F_{max} is the maximum friction force between the wheels of Yamabico-11 and the floor, to avoid slippage, we must have:

$$F_{max} \geq mv^2\kappa$$

or

$$\kappa_{max} = \frac{F_{max}}{mv^2} \tag{8.4}$$

3. Let a_{max} be the maximum acceleration attainable by Yamabico-11. Assuming constant traveling speed v. From Equation 8.1 and 8.2, we have:

$$\begin{aligned} a_{max} \geq \left|\frac{dv_{r,l}}{dt}\right| &= \left|\frac{d}{dt}([1 \pm W\kappa]v)\right| = \left|\pm Wv\frac{d\kappa}{dt}\right| \\ &= Wv\left(\frac{d\kappa}{ds}\right)\left(\frac{ds}{dt}\right) = Wv^2\dot{\kappa} \end{aligned}$$

or

$$\dot{\kappa}_{max} = \frac{a_{max}}{Wv^2} \tag{8.5}$$

These artificial constraints imposed on Yamabico-11 are considered in the problem statement of this thesis. Which means that the result of this thesis can be readily tailored for implementation on Yamabico-11.

C. DESIRABLE MOTION PATH CHARACTERISTICS

Given S and G, the problem statement of this thesis calls for intermediate S_0 and G_0, such that only the middle portion of the path from S_0 and G_0 need to be smooth. The initial portion of the path from S to S_0, and the final portion from G_0 to G can be just feasible which allows for reversals.

In the case of Yamabico-11, it can rotate about its center when it is stationary at $point(S)$ and $point(G)$. This in effect eliminates the requirement of S_0 and G_0. It is thus possible to directly plan a smooth path from $point(S)$ to $point(G)$.

However, it may be academically more interesting to impose the kinematic constraints on Yamabico-11 robot even when it is stationary. It would also allow the result of this thesis to be directly implementable on Yamabico-11 without any tailoring effort.

D. SPECIFIC COST FUNCTION

Recall that in the graph search layer, a specific cost function is required to be defined for each specific application.

The following cost function is defined for a path obtained from the sensor-oriented method for Yamabico-11:

$$cost = \int_S^G c(p)\, ds \tag{8.6}$$

where p is a point on the path, and,

$$c(p) = \begin{cases} c_1 & \text{if } w_{min} \le w(p) \le w_{max} \wedge \theta(p) \text{ is parallel to an obstacle wall } \wedge \kappa(p) = 0 \\ c_2 & \text{otherwise, where } c_2 > c_1 \end{cases}$$

In plain language, path segments that are parallel to obstacles walls and are within the sonar range are of lower cost. All other path segments, including cubic spiral turns, circular maneuver arcs, spine extensions and the straight path segments

involved in the end portions, are of higher cost because they do not facilitate position correction using sonar returns.

E. ADDITIONAL CONSIDERATIONS FOR IMPLEMENTATION

1. The disc-shaped robot $\mathcal{D}$ approximating Yamabico-11 should be grown by a safety bumper of Δr to accommodate dead reckoning errors. Even if the position and orientation correction capability is activated, the corrections will be carried out only periodically and dead reckoning errors will still occur in between two corrections.

2. The initial and final configurations are given explicitly per path planning query. Yamabico-11 cannot compute its initial configuration autonomously at the moment because the vision matching technique is still premature for implementation. Some other higher level knowledge representation of W may also be required for Yamabico-11 to recognize its whereabouts in W besides brute-force matching of the geometry of W. The manual input of the initial configuration can be expected to suffer higher level of error in accuracy than that of the dead reckoning capability of Yamabico-11. Therefore, the safety bumper must be padded up by another measurement-error bumper of Δe.

IX. CONCLUSION

Motivated by solving a specific motion planning problem for Yamabico-11, the problem is generalized into an academic research topic of this thesis of which the Yamabico-11 application is but a special case.

The results of this thesis can be applied to motion planning problems for any non-holonomic disc-shaped robots which are subject to kinematic constraints of κ_{max} and $\dot{\kappa}_{max}$. Although the results are inexact, they should be sufficient in applications where the target operating environment is an office-like environment. The results should also allow fast computation that will facilitate real-time applications. The motion paths planned will have a nice property that except at the end portions where reversals are usually inevitable, the majority of the middle portion will allow non-stopping and forward-moving motions. To the best of the author's knowledge, no other results have produced paths of these properties subject to κ_{max} and $\dot{\kappa}_{max}$ constraints.

Research contributions of this thesis include:

1. Formalization of the layered motion planning approach framework

   Although breaking down the motion planning problem into two or three layers has been informally considered by researchers, this thesis is the first in formally laying down the framework of four-layered motion planning approach. The goal of each layer, together with the interfaces between two adjacent layers, are defined explicitly. Different methods are applied using the layered approach which demonstrates the versatility of such a framework.

2. The novel sensor-oriented global path planning method

   Of all the available global path planning method, none builds a roadmap that facilitates sensor-oriented robots in carrying out position and orientation correction using sonar returns. This thesis proposes a novel sensor-oriented method which builds a roadmap with spines running parallel to obstacles as far as possible and within the sonar range. Yet it avoids becoming a wall-following algorithm by letting a cost function chooses shortcuts wherever more economically.

3. Investigation of smooth path of class C^2 subject to κ_{max} and $\dot{\kappa}_{max}$ constraints

   This thesis investigates the problem of planning smooth path of class C^2 subject to κ_{max} and $\dot{\kappa}_{max}$ constraints. Such paths are very appealing because they can be followed by many different types of rigid body robot without stopping. However, in most research papers, only C^1 paths of piecewise C^2 subject to κ_{max} constraint are investigated. The $\dot{\kappa}_{max}$ constraint has not been investigated to the best of the author's knowledge. Specific techniques of transforming a global roadmap into a smooth roadmap of class C^2 are discussed in the local smooth path planning layer.

4. Detailed study of the properties of cubic spirals

   A class of smooth curves called cubic spirals is adopted in this thesis for planning smooth C^2 paths. Previous studies on the properties of cubic spirals for path planning are made in isolated context, where obstacle collisions and kinematic constraints are not considered. Because of this, the elegant curve of cubic spiral could not find its way to applications where the full context of path planning are to be considered. This thesis investigates further properties of cubic spirals subject to κ_{max} and $\dot{\kappa}_{max}$ constraints. It fills in the missing

links for the cubic spiral by proposing using bounding regions BR_1 and BR_2 to approximate the trajectory of a cubic spiral which has no closed form. For the first time a practical approximate algebraic solution is available for collision checking against the cubic spiral.

5. Investigation of end portions path planning

   In many research papers, the end portions path planning problem is either avoided (by assuming S and G lie on the roadmap) or briefly mentioned without outlining the planning procedures. This problem is studied in detail in the end portions path planning layer, and a planning procedure is specified explicitly. Although there are still rooms for improvement, such effort is the first attempt in tackling the end portions path planning problem head on.

One important area that is open for future research is a good definition for path cost function. As pointed out in Chapter VII, there are many factors that can be included in a path cost function. In this thesis, only a composite cost function of wall-following distance versus shortness is proposed. Other important factors such as number of reversals, number of turns, etc, are ignored. It is still an open question on how to blend all these factors into a single uniform cost function.

Another interesting topic for future research is to extend the results of this thesis to polygon-shaped robots, for example, car-like vehicles.

LIST OF REFERENCES

[1] J. C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, Norwell, MA, 1991.

[2] Lozano-Pérez, T., Spatial Planning: A Configuration Space Approach, *IEEE Trans. Comp.* , 32 (2) (1983) 108–119.

[3] L. P. Chew, Planning the Shortest Path for a Disc in $O(n^2 \log n)$ Time, *Proceedings ACM Symposium on Computational Geometry*, (1985) 214–223.

[4] J. Laumond, Feasible Trajectories for Mobile Robots with Kinematic and Environment Constraints, *Intelligent Autonomous Systems*, (1986) 346–354.

[5] H. A. Vasseur, F. G. Pin, and J. R. Taylor, Navigation of a Car-like Mobile Robot Using a Decomposition of the Environment in Convex Cells, *Proceedings IEEE Int. Conf. on Robotics and Automation*, (1991) 1496–1502.

[6] S. Fortune and G. Wilfong, Planning Constrained Motion, *4th ACM Symp. on Computational Geometry*, (1988) 445–459.

[7] G. Wilfong, Motion Planning for an Autonomous Vehicle, *IEEE Int. Conf. on Robotics and Automation*, (1988) 529–533.

[8] P. Tournassoud and O. Jehl, Motion Planning for a Mobile Robot with a Kinematic Constraint, *IEEE Int. Conf. on Robotics and Automation*, (1988) 1785–1790.

[9] Th. Fraichard, Smooth Trajectory Planning for a Car in a Structured World, *IEEE Int. Conf. on Robotics and Automation*, (1991) 318–323.

[10] C. O'Dunlaing and C. Yap, A "Retraction" Method for Planning the Motion of a Disc, *J. Algorithms*, 6 (1985) 104–111.

[11] G. Wilfong, Shortest Paths for Autonomous Vehicles, *IEEE Int. Conf. on Robotics and Automation*, (1989) 15–20.

[12] Y. Kanayama and B. I. Hartman, Smooth Local Path Planning for Autonomous Vehicles, *IEEE int. conf. on Robotics and Automation*, (1989) 1265–1270.

[13] R. A. Brooks, Solving the Find-Path Problem by Good Representation of Free Space, *IEEE Trans. on Systems, Man and Cybernetics*, (1983) SMC–13(3) 190–197.

[14] Y. Kanayama and N. Miyake, Trajectory Generation for Mobile Robots, *Robotics Research, vol. 3, MIT Press*, (1986) 333–340.

[15] K. Komoriya and K. Tanie, Trajectory Design and Control of a Wheel-type Mobile Robot Using B-spline Curve, *IEEE/RSJ Int. Workshop on Intelligent Robots and Systems*, (1989) 398–405.

[16] P. Jacobs and J. Canny, Planning Smooth Paths for Mobile Robots, *IEEE Int. Conf. on Robotics and Automation*, (1989) 2–7.

[17] Y. Kanayama, Introduction to Spatial Reasoning, *Lecture Notes of the Advanced Robotics Course, Dept. of Computer Science, Naval Postgraduate School, Fall Quarter 1991.*

[18] Y. Kanayama and M. Onishi, Locomotion Functions in the Mobile Robot Language, MML, *IEEE int. conf. on Robotics and Automation*, (1991) 1110–1115.

[19] Y. Kanayama and Y. Shin'Ichi, Vehicle Path Specification by a Sequence of Straight Lines, *IEEE Journal of Robotics and Automation, vol. 4, no. 3*, (1988) 265–276.

[20] J. T. Schwartz and M. Sharir, A Survey of Motion Planning and Related Geometric Algorithms, *Artificial Intelligence*, 37 (1988) 157–169.

[21] M. Sharir and A. Schorr, On Shortest Paths in Polyhedral Spaces, *Tech. Report No. 138, Dept. of Computer Science, Courant Institute of Mathematical Sciences, New York, N. Y., October 1984.*

[22] J. Barraquand and J. C. Latombe, Nonholonomic Multibody Mobile Robots: Controllability and Motion Planning in the Presence of Obstacles, *IEEE Int. Conf. on Robotics and Automation*, (1991) 2328–2335.

INITIAL DISTRIBUTION LIST

1.	Dudley Knox Library Code 52 Naval Postgraduate School Monterey, CA 93943-5000	2
2.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
3.	Chairman, Computer Science Department Code CSMz Naval Postgraduate School Monterey, CA 93943-5000	1
4.	Professor Yutaka Kanayama Code CSKa Naval Postgraduate School Monterey, CA 93943-5000	3
5.	Professor Man-Tak Shing Code CASh Naval Postgraduate School Monterey, CA 93943-5000	1
6.	LCDR Donald P. Brutzman Code ORBr Naval Postgraduate School Monterey, CA 93943-5000	1
7.	Professor Masatoshi Shirayama Department of Computer Science Ishikawa National College of Technology Tsubata-Machi, Ishikawa 929-03 Japan	1

8. Chief Defence Scientist 1
HQ Ministry of Defence
MINDEF Building
Gombak Drive
Off Upper Bukit Timah Road
Singapore 2366
Republic of Singapore

9. Mr. Tan Liek Foo 1
Defence Science Organisation
Ministry of Defence, Singapore
20 Science Park Drive
Singapore 0511
Republic of Singapore

Thesis
T13569 Tan
c.1 Motion planning for
rigid body robots.

DUDLEY KNOX LIBRARY
3 2768 00034300 8